SONDERHEFT 18 DM 14,-

★HAPPY-COMPUTER★ 8. SCHNEIDER SONDERHEFT

★HAPPY-★ COMPUTER

DAS GROSSE HEIMCOMPUTER-MAGAZIN

Basic schlägt alles

- ★ So programmiert man Grafik
- ★ Kürzer, schneller, besser: Programmieren mit Plan
- ★ Referenzkarte: alle Befehle auf einen Blick

Disketten im Griff

- ★ Raffiniert programmiert
- ★ So geht's: Kopierschutz

Programmiersprachen

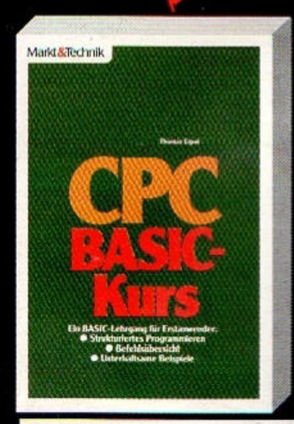
- ★ Die richtige Sprache für jeden
- ★ Große Marktübersicht

Super EPROMer selbst gebaut

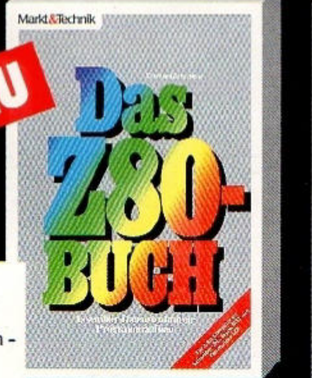
Spielespaß mit tollen Listings

Alle Programme auch auf Diskette erhältlich

Bücher zu Schneider CPCs und Joyce



T. Erpel
CPC-BASIC-Kurs
1985, 376 Seiten
Ein BASIC-Lehrgang für Erstanwender: strukturiertes Programmieren, Befehlsübersicht, unterhaltsame Beispiele.
Best.-Nr. MT 828
ISBN 3-89090-167-0
DM 46,–/sFr 42,30/öS 358,80



E. Zehendner
Das Z80-Buch
1987, 682 Seiten
Assembler – Datenstrukturen – Programmaufbau
Best.-Nr. 90219
ISBN 3-89090-219-7
DM 59,–/sFr 54,30/öS 460,20

C. Strausch
Schneider-CPC-Grafik-Programmierung
1986, 231 Seiten.
Die faszinierende Welt der Grafik, erklärt an zahlreichen Anwendungsbeispielen. Mit vielen Tips & Tricks: BASIC-Befehlserweiterung, Sprites, Hardcopy-Routinen.
Best.-Nr. 90182
ISBN 3-89090-182-4
DM 46,–/sFr 42,30/öS 358,80

J. Hückstädt
Textverarbeitung mit LocoScript
1986, 246 Seiten
Ein unentbehrliches Lehrbuch und Nachschlagewerk für jeden Joyce-Besitzer: Texte schreiben, aufbereiten und drucken.
Best.-Nr. 90198
ISBN 3-89090-198-0
DM 39,–/sFr 35,90/öS 304,20

W. Kassera
Turbo-Pascal auf dem CPC 6128
2. Quartal 1987. ca. 250 Seiten.
Ausführlich erläuterte Übungen, Beispiele und Anwendungen für das CP/M-Plus-System in über 100 Programmteilen, Prozeduren und Funktionen.
Best.-Nr. 90455
ISBN 3-89090-455-6
DM 59,–/sFr 54,30/öS 460,20

O. Hartwig
Experimente zur Künstlichen Intelligenz in BASIC auf CPC 464/664/6128
2. Quartal 1987, ca. 300 Seiten
Eine praxisbezogene Einführung in das Verarbeiten natürlicher Sprache. Wissensrepräsentation, Computer-Kreativität, Robotics und Expertensysteme.
Best.-Nr. 90473
ISBN 3-89090-473-4
DM 49,–/sFr 45,10/öS 382,20

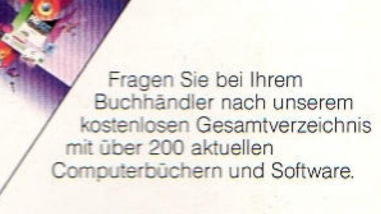
J. Hückstädt
CP/M-Plus-Anwenderhandbuch CPC 6128/Joyce
1986, 256 Seiten.
Ein unentbehrliches Nachschlagewerk für die praktische Arbeit mit CP/M Plus und seinen Hilfsprogrammen. Mit zahlreichen Beispielen und ausführlichen systemspezifischen Daten zur internen Speicherorganisation und zu Schnittstellen.
Best.-Nr. 90197
ISBN 3-89090-197-2
DM 46,–/sFr 42,30/öS 358,80

Markt&Technik-Produkte erhalten Sie bei Ihrem Buchhändler, in Computerfachgeschäften oder in den Fachabteilungen der Warenhäuser.

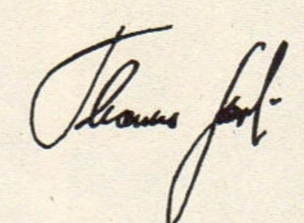
Irrtümer und Änderungen vorbehalten.

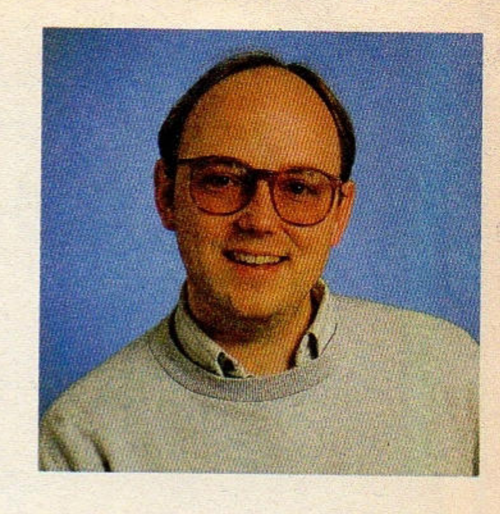
Markt&Technik
Zeitschriften · Bücher
Software · Schulung

Markt&Technik Verlag AG, Buchverlag, Hans-Pinsel-Straße 2, 8013 Haar bei München, Telefon (089) 4613-0.
SCHWEIZ: Markt&Technik Vertriebs AG, Kollerstrasse 3, CH-6300 Zug, Telefon (042) 415656,
ÖSTERREICH: Rudolf Lechner&Sohn, Heizwerkstraße 10, A-1232 Wien, Telefon (0222) 677526, Ueberreuter Media Handels- und Verlagsges. mbH (Großhandel), Alser Straße 24, A-1091 Wien, Telefon (0222) 481538-0



Fragen Sie bei Ihrem Buchhändler nach unserem kostenlosen Gesamtverzeichnis mit über 200 aktuellen Computerbüchern und Software.





Computerspaß nach Maß

Warum gibt es von Happy-Computer nur Sonderhefte zu den Schneider-Computern und kein eigenes Magazin? Solche oder ähnliche Fragen unserer Leser erreichen uns, fast schon seit der erste CPC im Jahr 1984 auf dem Markt erschien. Und natürlich machten wir uns auch Gedanken über das Informationsbedürfnis der stolzen Besitzer dieses erfolgreichen Newcomers. Wir entschieden uns damals, wie inzwischen wohl ein jeder weiß, für eine regelmäßige, im dreimonatigen Rhythmus erscheinende Sonderheftreihe. Damit, so glauben wir, bieten wir Ihnen als aktivem CPC-Benutzer eine gelungene Mischung vielfältigster Informationen, Tips und Listings. Unser monatliches Magazin Happy-Computer bringt Ihnen einen hochaktuellen Überblick über den Gesamtmarkt und generelle technologische Entwicklungen auf dem Computersektor. Diesem äußerst wichtigen Anspruch kann man in einem reinen »Schneider-Magazin« leider nicht gerecht werden. Aber schließlich haben Sie ein berechtigtes Interesse daran, daß Ihr Blick nicht durch »Scheuklappen« begrenzt wird.

Dazu kommen die neuesten, auch speziell aus der Schneider-Welt stammenden Neuheiten, die Sie oft bei uns zuerst erfahren, bevor andere davon auch nur Gerüchte kennen.

Nichtsdestotrotz gibt es einen eigenen Schneider-Teil mit aktuellsten Hard- und Software-Tests, Tips, Kursen und ausgesuchten Spitzen-Listings. Nehmen Sie all diese interessanten Themen zusammen, also den Schneider-»Teil«, die umfassende Markt-Übersicht (durch Messeberichterstattungen von allen wichtigen Computermessen dieser Welt, Tests von Geräten sämtlicher Hersteller und Trendberichten), die allgemeinen Grundlagen (beispielsweise über Algorithmen zur Grafik-Programmierung, für Sortier-Routinen, und, und, und...), den Spiele-Sonderteil und dergleichen mehr, haben Sie einen Informationsumfang für CPC-Besitzer, der seinesgleichen sucht.

Umfangreiche Programmlistings und Grundlagenbeiträge packen wir dann in konzentrierter Form in unsere Schneider-Sonderhefte, die Ihnen lange Zeit als Nachschlagewerke zur Verfügung stehen. Damit decken wir alle Anwendungsgebiete und Interessen des Computer-Hobbyisten ab.

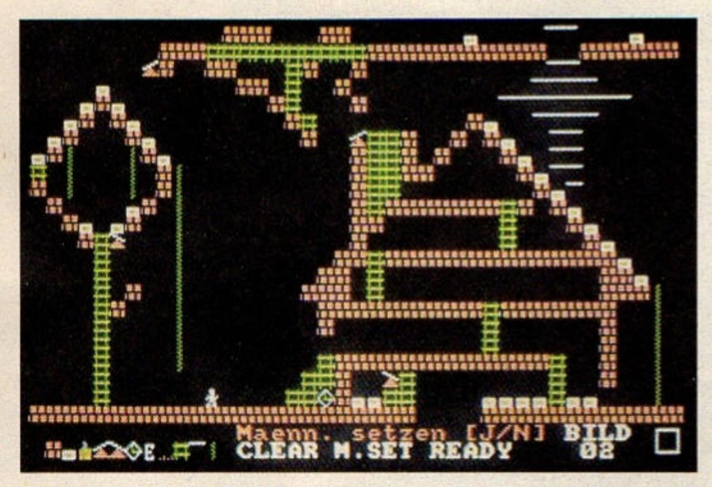
Diese Ausgabe bietet in einem Disketten-Schwerpunkt den Vergleichstest dreier Kopierprogramme. Eigentlich sollte ein ähnlicher Test von Kassetten-Kopierern das Thema ergänzen. Aber schon erste Versuche im Vorfeld führten dieses Vorhaben ad absurdum, da jedes der Kopierprogramme nur eine »Handvoll« anderer Programme kopierte. Die Moral von der Geschicht: Greifen Sie zum Kopieren von Kassettensoftware zu einem der ab Seite 11 getesteten Module.

Spielefans dürfen sich schon jetzt besonders auf das nächste Schneider-Sonderheft freuen, denn einen Schwerpunkt dieser nächsten Ausgabe werden neben den gewohnten Rubriken die besten Spielelistings bilden. Da wir schon in diesem Moment, wo Sie diese Zeilen lesen, mit Hochdruck an der Produktion der kommenden Ausgabe arbeiten, sollten Sie uns schnell Ihre Programme zusenden. Vielleicht liegt auch bei Ihnen ein »Juwel« in der Schublade, über dessen Veröffentlichung die anderen Leser hocherfreut wären.

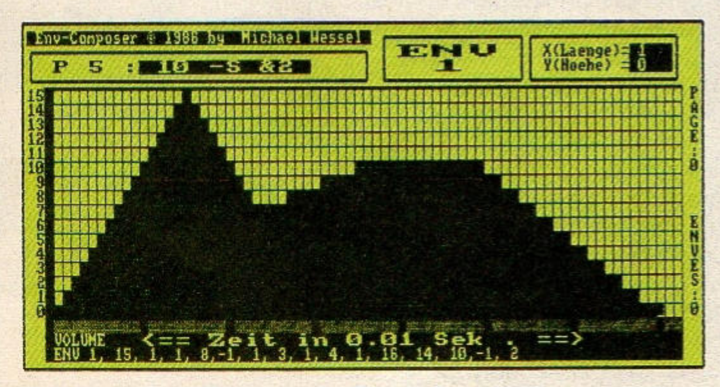
Thomas Jacobi

Inhalt

Wer hat davon nicht schon geträumt: Programme im EPROM ersparen Ladezeiten und belegen kaum Arbeitsspeicher. Unser Selbstbau-EPROMer macht mit intelligenter Software und geringen Kosten Träume wahr. 14



Für die nötige Entspannung bei der Arbeit mit dem CPC sorgt das flotte Spiel »Stone Runner«, bei dem es auf Überlegung und flinke Finger ankommt. Ein Spielfeld-Editor macht den Spielspaß komplett. 128



Ein reines Vergnügen war der Entwurf von Hüllkurven bislang fürwahr nicht. Mit dem »Envelope Composer« gelingen Ihnen zukünftig selbst komplizierteste Töne schnell und komfortabel am Bildschirm. 91

Hardware

Bastelei

Grundlagen

Software

Anwendungs-Listing

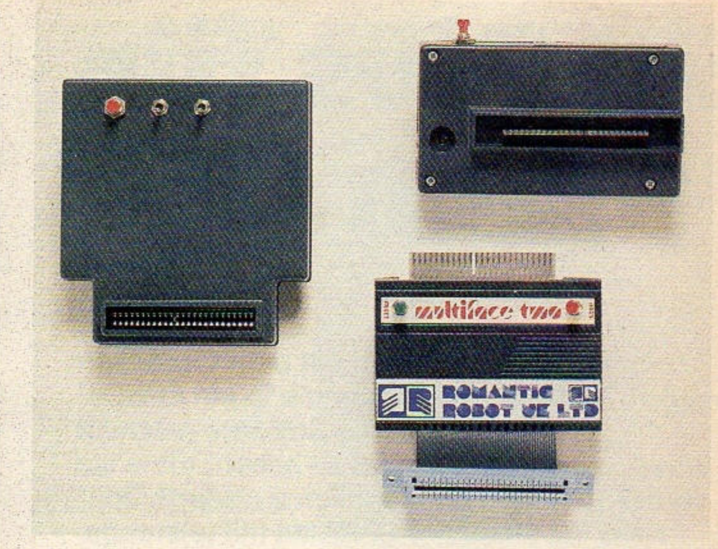
Utility-Listing

Spiele-Listing

Grafik-Listing

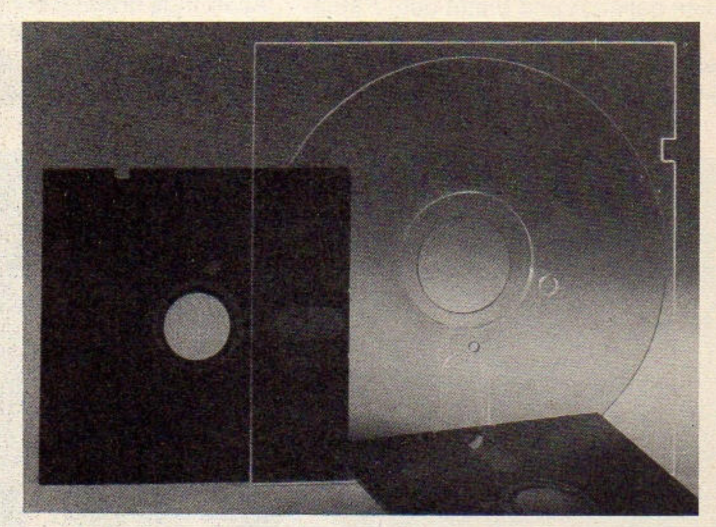
Tips & Tricks

Sonstiges



Drei Kopiermodule sind mittlerweile für den CPC erhältlich. Manch einer zögert angesichts der hohen Preise und der fehlenden Vergleichsmöglichkeiten. Wir haben uns für Sie alle drei angesehen und geben Entscheidungshilfen. 11

Die sogenannten »Apfelmännchen«-Grafiken hat wohl schon jeder Computerinteressierte bestaunt. Wenn Sie wissen wollen wie sie entstehen, lesen Sie die mathematischen Grundlagen und Sie werden sehen: es ist nicht kompliziert. 140



Jeder hat sie schon einmal gesehen. Die meisten benutzen sie bereits. Was aber beim Gebrauch einer Diskette genau passiert, wissen die wenigsten. Lernen auch Sie, mehr aus diesem Speichermedium herauszuholen. 31

Ein Plus für jeden CPC

Haben Sie auch schon mit dem Gedanken gespielt, Ihre besten Programme in ein EPROM zu brennen, um die Software jederzeit beim Programmieren parat zu haben, ohne sie vorher jedesmal umständlich zu laden? Oder möchten Sie eines der EPROMs mit leistungsfähiger kommerzieller Software, wie sie mittlerweile vermehrt angeboten werden, kaufen und in Ihrem Computer professionell einsetzen?

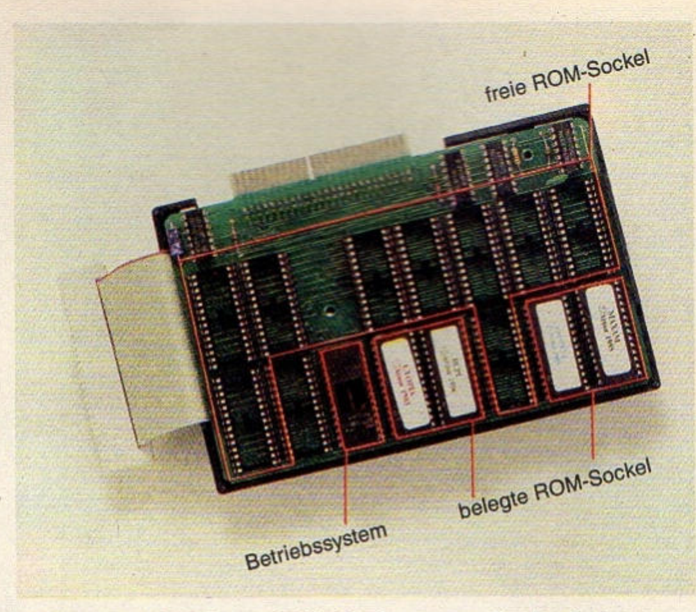


Bild 2. Die ROM-Box mit fünf belegten und zehn freien EPROM-Steckplätzen

EPROMs kann man mit EPROM-Brennern selbst programmieren (lesen Sie hierzu auch die Bauanleitung in dieser Ausgabe) oder aus dem inzwischen reichhaltigen Angebot der Softwarehäuser wählen. Doch wie ein EPROM an den CPC anschließen? Wie läßt sich die neue Software in das Betriebssystem einbinden?

Man muß weder Bastelprofi noch Betriebssystemspezialist sein, um diese Frage zu lösen, denn mittlerweile gibt es Abhilfe in Form einer Hardware-Erweiterung für den CPC. »Super ROM Plus Box« (im folgenden kurz ROM-Box genannt) heißt das hochinteressante Produkt von Britannia Developments, das seit kurzem auch in Deutschland erhältlich ist.

Die ROM-Box wird in einem farblich zum CPC passenden Kunststoffgehäuse geliefert und bietet dem Käufer bei einem Preis von 149 Mark 15 Steckplätze für ROMs und EPROMs (16-KByte-Typen), von denen einer bereits durch das Betriebssystem der ROM-Box belegt ist. (Der Einfachheit halber werden wir im folgenden auch EPROMs als ROMs bezeichnen.)

Bild 1 zeigt, wie die ROM-Box am CPC angeschlossen wird, und auf Bild 2 erkennt man das geöffnete Gehäuse mit fünf ROMs auf der Platine.

Die ROM-Box arbeitet mit allen drei CPC-Modellen zusammen und wird auf den Erweiterungsanschluß aufgesteckt. Beim CPC 6128 ist zusätzlich ein Adapter für 30 Mark erforderlich. Sämtliche Signale des Erweiterungsanschlusses sind auf einen zweiten Stecker herausgeführt, so daß bereits vorhandene Erweiterungen weiterhin betrieben werden können. Auch hier ist beim CPC 6128 wieder ein Adapter nötig, wenn die Peripherie über einen 50poligen Amphenolstecker verfügt. Allerdings arbeitet die ROM-Box nicht mit den Produkten von Vortex zusammen, weder mit den Laufwerken noch mit der Speichererweiterung.

Sämtliche ROMs in der Box werden beim Einschalten des Computers initialisiert und in das Betriebssystem des CPC eingebunden, sofern sie den Amstrad-Spezifikationen für Erweiterungs-ROMs entsprechen. Die Tabelle zeigt, wie ein Erweiterungs-ROM aufgebaut sein muß, damit es das Betriebssystem des CPC auch korrekt initialisieren und einbinden kann.

Das Betriebssystem der ROM-Box übernimmt nach der Initialisierung der ROMs die Kontrolle über den Computer und zeigt auf dem Bildschirm ein Menü, das sich aus vier Fenstern zusammensetzt (Bild 3).

Im ersten Fenster sind alle Vordergrund-ROMs (zum Beispiel das Basic des CPC) aufgeführt, die sich auf Tastendruck auswählen und aktivieren lassen. Das zweite Fenster zeigt die Hintergrund-ROMs (zum Beispiel das Disketten-Betriebssystem), die sich auf Tastendruck ein- und ausblenden lassen.

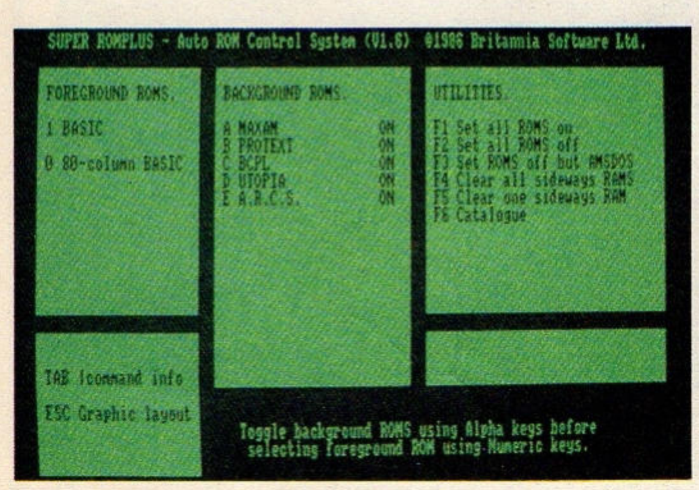


Bild 3. Das Menü des ROM-Box-Betriebssystems, das nach Einschalten des Computers erscheint

GRAPHIC LAYOUT OF SUPER ROMPLUS

This diagram shows physical layout of all sockets and displays both socket number and contents.

N.B. empty sockets may appear occupied due to ROMS fitted in other peripherals such as CPM ROM, etc.

9 **** EMPTY ****
10 **** EMPTY ****
11 **** EMPTY ****
12 **** EMPTY ****
13 **** EMPTY ****
14 **** EMPTY ****
15 **** EMPTY ****
1 MAXAM
2 PROTEXT
3 **** EMPTY ****
4 BCPL
5 UTOPIA
6 A.R.C.S.
7 **** EMPTY ****
8 **** EMPTY ****

Bild 4. Ein Unterpunkt des Menüs untersucht die ROM-Box auf freie und belegte Steckplätze

Byte	Inhalt
0	Kennung für Speichertyp (128=Vordergrund-ROM, 1=Hintergrund-ROM)
1	Vorkommateil der Versionsnummer (ohne Einfluß)
2	Nachkommateil der Versionsnummer (ohne Einfluß)
3	Modifikationsnummer (ohne Einfluß)
4	Unteres Byte der Adresse der Befehlswort-Tabelle
5	Oberes Byte der Adresse der Befehlswort-Tabelle
6	Sprung (JP) zur Routine für das erste Befehlswort in der Befehlswort-Tabelle (in der Regel eine Initialisierungsroutine, weil 6 auch der Einstiegspunkt bei der Initialisierung durch das Betriebssystem ist)
9	Sprung (JP) zur Routine für das zweite Befehlswort in der Befehlswort-Tabelle
12 : :	Sprung (JP) zur Routine für das dritte Befehlswort : :
n	Sprung (JP) zur Routine für das letzte Befehlswort
n+1	Befehlswort-Tabelle (jedes Befehlswortende wird duch ein gesetztes achtes Bit im letzten Buchstaben gekennzeichnet)
m	Befehlswort-Tabellenende (0)
m+1 : :	Initialisierungs-Routine und Routinen für die einzelnen Befehlswörter : :

Nach diesem Schema muß ein Erweiterungs-ROM aufgebaut sein, damit der CPC den Baustein initialisieren und einbinden kann

Bild 1. Die »Super-ROM-Box« präsentiert sich farblich abgestimmt zum CPC

Durch die ESC-Taste läßt sich eine Übersicht, die die Belegung der Steckplätze in der ROM-Box zeigt, aufrufen (Bild 4).

Als besonderes Extra erlaubt die ROM-Box auch den Einsatz von statischen RAM-Bausteinen (sogenannte Sideways-RAMs) als Speichererweiterung. Das Betriebssystem kann diese RAMs als Druckerpuffer verwenden, um Wartezeit beim Ausdruck von Programmen oder Texten zu sparen oder von Diskette beziehungsweise Kassette mit Daten laden, so daß jedes Sideways-RAM als ROM-Emulator zum Austesten von Software, die später in einen EPROM gebrannt werden soll, geeignet ist.

Das Betriebssystem der ROM-Box enthält neben den Routinen zur Verwaltung der ROMs zusätzlich 20 leistungsfähige RSX-Befehle, die aus der Basic-Ebene heraus aufgerufen werden können. Drei Befehle listen allgemeine Informationen zu allen vorhandenen ROMs oder zeigen detaillierte Informationen zu einem einzelnen ROM an. Drei weitere Befehle dienen zur Verwaltung der Sideways-RAMs in der ROM-Box, ein RSX-Befehl paßt CP/M Plus an das ROM-Box-Betriebssystem an. Die restlichen 13 RSX-Befehle sind für die Ansteuerung von (Epson-kompatiblen) Druckern zuständig, um Schriftart und Vorschub einzustellen.

Im Lieferumfang der ROM-Box ist eine ausführliche englische und deutsche Anleitung enthalten. Sind Sie jedoch im Englischen etwas bewandert, werden Sie mit der Bedienung der ROM-Box ohnehin keine Probleme haben, weil das Betriebssystem der ROM-Box genügend Hinweise zur Bedienung gibt.

Zusammenfassend beurteilt, erhält man mit der ROM-Box für den CPC eine leistungsfähige Erweiterung zu einem fairen Preis. Sie bietet durch das eingebaute Betriebssystem eine komfortable Verwaltung von gekauften und selbstprogrammierten EPROMs und erweitert dadurch die Fähigkeiten des CPC beträchtlich. Mit Ihren »Lieblings-EPROMs« bestückt, werden Sie die ROM-Box schon nach kurzer Zeit nicht mehr missen wollen.

(ma)

PR8-Soft, Klaus-M. Pracht, Erbachshof 1, 8702 Eisingen, Tel. (09306) 8735

Kampf der Speicher-Giganten

RAM-Erweiterungen zweier Hersteller im Vergleichstest: Welche bietet den optimalen Nutzen?

It's British – it won't work. Dieser scherzhafte Ausspruch amerikanischer Techniker hat in der jetzigen Computer-Generation eher einen Bumerang-Effekt für seine Erfinder. Gerade die findigen Engländer zeigen nämlich deutlich, daß sie sehr gut in der Elektronikentwicklung und der Programmierung Bescheid wissen. So zählte zu den ersten Firmen, die Hard- und Software für den CPC anboten, die britische Firma dk'tronics. Sie ist in erster Linie in ihrer Heimat bekannt, wo sie ihre Produkte vornehmlich verkauft. Zu diesen Produkten, die nun seit einiger Zeit auch auf dem deutschen Markt erhältlich sind, gehören 64- und 256-KByte-Speichererweiterungen für den CPC 464 und 664 sowie weitere Hardware-Erweiterungen für alle CPCs mit eigenem 64- oder 256-KByte-Speicherbereich, der als virtuelles Diskettenlaufwerk anzusprechen ist (Silicon Disc).

Auf dem Gebiet der Speichererweiterungen war eine deutsche Firma schneller: Vortex entwickelte eine auf 512 KByte aufrüstbare 256-KByte-Erweiterung, die allerdings nur mit den CPC-Modellen 464 und 664 zu betreiben ist. Dafür vereint sie Fähigkeiten, die man bei dk'tronics nur durch Kombination der Speichererweiterung mit der Silicon Disc erhält.

Die Rollen in diesem Vergleichskampf sind klar verteilt: Auf der einen Seite steht die Vortex-Erweiterung, die in Deutschland einige Unterstützung seitens der Software-Industrie findet. Auf der anderen Seite steht der hierzulande relativ unbekannte Herausfor-

derer von der britischen Insel, der dort ein hohes Ansehen genießt. Wer macht das Rennen?

Eine Speichererweiterung funktioniert natürlich nicht ohne unterstützende Software, denn der Prozessor der CPCs, ein Zilog Z80, adressiert bekanntlich nur einen Speicherbereich von 64 KByte direkt. Und so bietet die dk'tronics-Erweiterung unter Basic 12 RSX-Befehle, die zum Beispiel das schnelle Speichern und Laden von Bildschirminhalten als Windows im zusätzlichen RAM-Bereich erlauben. Die Bilder in den einzelnen Speicherbänken sind auch blitzschnell zur Anzeige auf dem Bildschirm auszutauschen, um trickfilmähnliche Effekte zu erzielen. Durch Variablenablage im zusätzlichen Speicherbereich spart man kostbaren Platz im normalen Arbeitsspeicher, der dadurch umfangreichere Basic-Programme faßt. Damit ähneln die Fähigkeiten eines CPC 464 oder 664 mit dieser Erweiterung denen des CPC 6128. Die maximale Menge der Bilder und Daten hängt selbstverständlich von der Größe der Erweiterung ab. Der Nutzen der Speichererweiterung unter Basic läßt sich durch gemeinsamen Betrieb mit der Silicon Disc steigern. Das Betriebssystem der Silicon Disc macht deren RAM-Bereich nämlich als Pseudo-Diskettenlaufwerk mit einer maximalen Kapazität von 444 KByte nutzbar, was beim häufigen Wechsel zwischen mehreren Programmen, intensivem Datentransfer sowie beim Kopieren wesentliche Geschwindigkeitsvorteile mit sich bringt. Der Zugriff auf Daten in der RAM-Disk geschieht dermaßen schnell, daß Ladezeiten kaum mehr ins Gewicht fallen.

Vortex dagegen bietet 66 neue Befehle. Mit der neuesten Version des Vortex-BOS (»Bank Operating System«) steht wie bei dk'tronics unter Basic eine RAM-Disk mit ebenfalls 444 KByte (im Vollausbau) bereit. Die neuen Befehle zur relativen Dateiverwaltung erlauben völlig neue Datenstrukturen.

Vortex SP 256 – ein wahres Füllhorn

Auch die Fähigkeit, in komfortabelster Weise Bildschirme zu speichern und eine Art Trickfilm ablaufen zu lassen, fehlt hier nicht.

Ein im Umfang variabler Drucker-Spooler sorgt dafür, daß man selbst beim Druck langer Texte am Computer normal weiterarbeiten kann, während der Drucker noch fleißig rattert.

Bild 1. Die dk'tronics-Silicon-Disc besteht in der 256-KByte-Version aus zwei Modulen; dem Betriebssystem und dem Speicher, während die Arbeitsspeicher-Erweiterung in einem Gehäuse Platz findet.

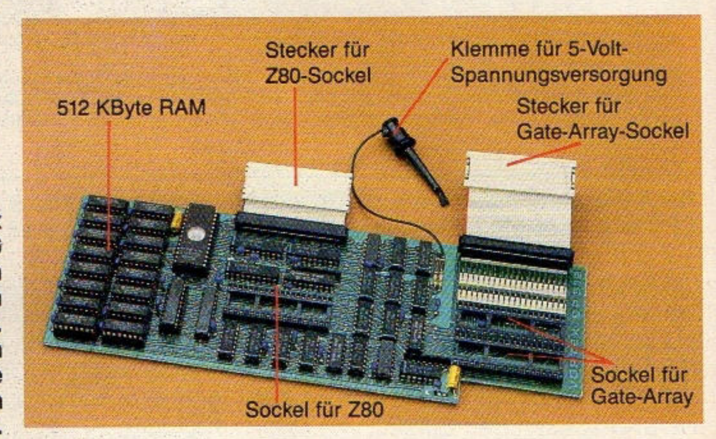


Bild 2. Die Vortex SP 512 und SP 256 werden als Platinen zum Einbau in den Computer geliefert. Steckverbindungen reduzieren die »Bastelei« jedoch auf ein Minimum.

Einzigartig ist die Fähigkeit der Verwaltung größerer Basic-Programme, deren Programmcode einen Umfang von bis zu 320 KByte erreichen darf.

Eine besonders interessante der zusätzlichen Funktionen ist das eingebaute Monitorprogramm. Mit ihm lassen sich nicht nur kleine Maschinencode-Routinen schreiben, sondern auch Programme vom Datenträger einlesen, verändern und anschließend speichern.

Viele zusätzliche nützliche Befehle runden das Gesamtbild der Ausstattung ab.

Beide Speichererweiterungen bieten auch unter CP/M 2.2 wahlweise einen vergrößerten Programmspeicher (TPA) von 61 KByte (dk'tronics) beziehungsweise 62 KByte (Vortex) anstelle der normalen 43 KByte. Umfangreiche professionelle CP/M-Programmpakete arbeiten oft nur mit diesem größeren Arbeitsspeicher. Dazu zählen Programme wie dBase II und Multiplan. Andere, wie beispielsweise Wordstar oder Turbo-Pascal, um nur zwei bekannte Namen zu nennen, arbeiten zwar unter dem »kleinen« CP/M 2.2 zufriedenstellend, gelangen jedoch erst mit der größeren TPA zu ihrer vollen Leistungsfähigkeit.

Sowohl die Silicon Disc als auch die Vortex-Erweiterung bieten die Vorzüge ihrer RAM-Disks auch unter CP/M. Dazu kommt bei Vortex noch der Drucker-Spooler.

Dafür bieten die Speichererweiterungen von dk'tronics unter CP/M andere Vorzüge. Sie machen nämlich auf den kleinen CPC-Modellen 464 und 664 auch CP/M Plus verfügbar – vorausgesetzt, man hat einen CPC 6128 als Zweitcomputer, denn das Kopieren der CP/M-Plus-Systemdiskette des Freundes ist selbstverständlich illegal. Hat man aber diese Klippe umschifft, steht der Freude über den neuen Bedienungskomfort dieses Betriebssystems nichts mehr im Weg.

In der Handhabung recht unterschiedlich

Ein sehr wichtiges Vergleichskriterium ist die Bedienungsfreundlichkeit. Die Entwickler bei dk'tronics sind davon ausgegangen, daß der durchschnittliche Anwender nur ungern seinen heißgeliebten Computer auseinanderschraubt. Die Erweiterungen stecken deshalb in eigenen Gehäusen (Bild 1), die ihren Platz am Expansion-Port des CPC finden. In Fällen mangelnder Software-Kompatibilität braucht man sie nur abzuziehen und hat im Nu einen ganz normalen CPC. Am CPC 464 können Probleme auftreten, für die jedoch die Amstrad-Entwickler die Verantwortung tragen, nicht etwa dk'tronics. Die Rede ist vom berüchtigten »CAT-Syndrom«. Für unsere Leser, die es noch nicht kennen: Bei Verwendung eines externen Diskettenlaufwerks wird die Einschaltmeldung nicht korrekt ausgegeben

Happy-Computer Sonderheft

PROGRAMM-SERVICE

Top-Listings dieser Ausgabe:

ANWENDUNGEN **Shares** Verwaltet Ihren privaten Aktienbestand und gibt Ihnen detaillierte Auskunft über die Kursentwicklung. **Multi-RAM-Disk** Ein Datenverwaltungsprogramm exklusiv für Benutzer einer Vortex-Speichererweiterung. Nutzt den zusätzlichen Speicherplatz optimal. **Boxenkit** Viele bauen sich heute ihre Lautsprecherboxen selbst. Die zeitaufwendigen und komplizierten Berechnungen für Frequenzweichen, Gehäusevolumen etc. übernimmt jetzt Ihr CPC.
SPIELE **Stone-Runner** Ein Kletter- und Geschicklichkeitsspiel der Extraklasse. Zu den neun integrierten Levels entwerfen Sie mit Hilfe des ebenfalls vorhandenen Editors beliebig viele eigene. Eine Highscore-Tabelle speichert Ihre Erfolge dauerhaft. **Spacetrap** Dieses Spiel entsteht im Laufe unseres Kurses zur stilistisch besseren Basic-Programmierung. Trotzdem oder gerade deswegen ist es ein vollwertiges, schnelles Action- und Strategiespiel, das auch durch seine optische Gestaltung anspricht.
UTILITIES **Super-Utility** Der erste Teil enthält einen komfortablen Disketten-Datei-Editor für leichte Eingriffe in fertige (auch professionelle) Programme.
GRAFIK **Apfelmännchen** Die faszinierenden Grafiken erzeugt ein relativ kurzes Programm in Hisoft-Pascal. Damit Sie die Effekte auch ohne den Hisoft-Compiler nutzen können, enthält die Diskette neben dem Pascal-Quellisting auch den compilierten Maschinencode, der selbständig auf jedem CPC lauffähig ist.
Und natürlich alle anderen Programme aus dieser Ausgabe.

1 Diskette für Schneider-Computer
Bestell-Nr. 25718 **(sFr 29,50/öS 349,-*)** **DM 34,90***

Weitere Angebote zum Thema Schneider-Computer

Happy-Computer, Ausgabe 3/87

Happy-Painter: Super-Malprogramm mit vorbildlicher Bedienerführung und fantastischen Fähigkeiten für alle CPCs (Listing des Monats 1/87). **Disccopy:** Kopiert nahezu alle 3-Zoll-Disketten. Selbst »überlange« Spuren mit zehn Sektoren oder illegale Sektornummern stellen kein Problem dar. **Copyit:** Auch Besitzern eines Kassettenrecorders als Speichermedium steht mit Copyit ein leistungsfähiges Backup-Programm zur Verfügung. **Discservice:** Völlig neue und überaus praktische Funktionen. **Bruch:** Findet und zeigt sämtliche REMarks in Basic-Listings auf Bildschirm oder Drucker.
Und alle weiteren Programme aus den Happy-Computer-Ausgaben 1, 2 und 3/87 für die Schneider CPCs.

Diskette für Schneider-Computer

Bestell-Nr. 21703

DM 29,90* **sFr 24,90/öS 299,-***

Happy-Computer, Ausgabe 12/86

Goldrain. Wertet Ihre Spielkarten des Bild-Goldregen-Spiels aus. **Screen-Compressor.** Speichert Bildschirminhalte platzsparend und mit erheblichem Geschwindigkeitsgewinn. Sie haben dabei die Wahl zwischen ganzen Bildschirmen, Ausschnitten und Windows. **Kursiv.** Ideal für Textverarbeitung: Verwenden Sie auf dem Bildschirm denselben kursiven Zeichensatz wie auf dem Drucker. **Super-CLS.** Neuer RSX-Befehl zur effektvollen Bildschirmlöschung. **Newgosub.** Ein Patch des GOSUB-Befehls erlaubt strukturierte Basic-Programmierung mit Unterprogrammnamen (nur CPC 464). **DEC$-Patch.** Endlich die perfekte Abhilfe für einen Fehler im Basic-Interpreter des CPC 464: Die Syntax des Befehls DEC$ ist nun korrigiert und somit kompatibel zu den beiden anderen CPC-Modellen (nur CPC 464). **Public-Domain.** Als besonderen Leckerbissen bieten wir Ihnen verschiedene Public-Domain-Programme. Darunter finden Sie je einen Interpreter der KI-Sprachen Lisp und Prolog mit Dokumentation und Beispielen sowie einen Forth-Compiler und einen Makroassembler.

1 Diskette für Schneider-Computer
Bestell-Nr. LH 8612 SD
DM 34,90*/sFr 29,50/öS 349,-*

2 Kassetten für Schneider-Computer
Bestell-Nr. LH 8612 SK
DM 34,90*/sFr 29,50/öS 349,-*

*inkl. MwSt. Unverbindliche Preisempfehlung.

Programme aus früheren Happy-Ausgaben

Ausgabe	Thema	Bestell-Nr.		DM	sFr	öS
5/87	Schneider	20705	Diskette	29,90*	24,90	299,-*
3/87	Schneider	21703	Diskette	29,90*	24,90	299,-*
12/86	Schneider	LH 8612 SD	Diskette	34,90*	29,50	349,-*
		LH 8612 SK	2 Kassetten	34,90*	29,50	349,-*
11/86	Schneider	LH 8611 SD	Diskette	34,90*	29,50	349,-*
		LH 8611 SK	Kassette	34,90*	29,50	349,-*
9/86	Schneider	LH 8609 SD	Diskette	34,90*	29,50	349,-*
		LH 8609 SK	Kassette	34,90*	29,50	349,-*
7/86	Schneider	LH 8607 SD	Diskette	34,90*	29,50	349,-*
4/86	Schneider	LH 8604 SD	Diskette	29,90*	24,90	299,-*
		LH 8604 SK	Kassette	29,90*	24,90	299,-*
12/85	Schneider	LH 8512 D	Diskette	34,90*	29,50	349,-*
		LH 8512 G	Kassette	29,90*	24,90	299,-*

Programme aus früheren Happy-Sonderheften

Ausgabe	Thema	Bestell-Nr.		DM	sFr	öS
16/87	Schneider	25716	Diskette	34,90*	29,50	349,-*
		26716	Disk. (Demo Giga-CAD)	34,90*	29,50	349,-*
		27716	2 Disk. im Paket	49,80*	43,50	498,-*
13/87	Schneider	25713	Diskette	34,90*	29,50	349,-*
		26713	Kassette	34,90*	29,50	349,-*
10/86	Schneider	LH 86S10 D	Diskette	34,90*	29,50	349,-*
		LH 86S10 K	2 Kassetten	34,90*	29,50	349,-*
7/86	Schneider	LH 86S7 SD	Diskette	34,90*	29,50	349,-*
		LH 86S7 SK	Kassette	34,90*	29,50	349,-*
4/86	Schneider	LH 86S4 D	Diskette	34,90*	29,50	349,-*
		LH 86S4 K	Kassette	29,90*	24,90	299,-*
1/86	Schneider	LH 86S1 D	Diskette	34,90*	29,50	349,-*
		LH 86S1 K	Kassette	29,90*	24,90	299,-*
2/85	Schneider	LH 85S2 D	Diskette	34,90*	29,50	349,-*
		LH 85S2 V	5¼"-Diskette	34,90*	29,50	349,-*
		LH 85S2 K	Kassette	29,90*	24,90	299,-*

Einige Tips zum Umgang mit den Leserservice-Disketten:

Auf der Diskette zu dieser Ausgabe finden Sie ein Basic-Programm namens »README.BAS«. Da es am Anfang gespeichert ist, starten Sie es bitte zuerst. Sie erhalten dadurch Informationen über die enthaltenen Programme. Dort erfahren Sie zu jeder Datei, was sie bewirkt und wo der gedruckte Beitrag dazu in der Ausgabe zu finden ist.
Bei früheren Ausgaben hieß dieses Inhaltsverzeichnis ebenso beziehungsweise »LISTME.BAS«. Dort besteht es aus einer ASCII-Datei, die Sie mit »LOAD "README"« im normalen Locomotive-Basic laden und durch »LIST« auf den Bildschirm beziehungsweise mit »LIST #8« auf dem Drucker ausgeben.

Bestellungen bitte an: Markt & Technik Verlag AG, Unternehmensbereich Buchverlag, Hans-Pinsel-Straße 2, D-8013 Haar, Telefon (089) 4613-0. **Schweiz:** Markt & Technik Vertriebs AG, Kollerstrasse 3, CH-6300 Zug, Telefon (042) 415656. **Österreich:** Ueberreuter Media Handels- und Verlagsgesellschaft mbH (Großhandel), Alser Straße 24, A-1091 Wien, Telefon (0222) 481538-0, Microcomput-ique E. Schiller, Fasangasse 24, A-1030 Wien, Telefon (0222) 785661, Bücherzentrum Meidling, Schönbrunner Straße 261, A-1120 Wien, Telefon (0222) 833196. **Bestellungen aus anderen Ländern bitte nur schriftlich an:** Markt & Technik Verlag AG, Abt. Buchvertrieb, Hans-Pinsel-Straße 2, D-8013 Haar, und gegen Bezahlung einer Rechnung im voraus.

Bitte verwenden Sie für Ihre Bestellung und Überweisung die eingeheftete Postgiro-Zahlkarte, oder senden Sie uns einen Verrechnungs-Scheck mit Ihrer Bestellung. Sie erleichtern uns die Auftragsabwicklung, und dafür berechnen wir Ihnen keine Versandkosten.

und der Computer »hängt«, oder die Meldung »Press PLAY then any key:« erscheint. Oft ist auch das Directory beim Basic-Befehl CAT bis zur Unkenntlichkeit zerstückelt. Meist tritt das CAT-Syndrom bei Computern auf, an denen mehrere Hardware-Erweiterungen betrieben werden. Ursache für dieses ebenso störende wie auch unnötige Phänomen ist der ungepufferte Bus, über den der gesamte Datentransfer zwischen Computer und Erweiterungen am Expansion-Port läuft. Abhilfe bringt der Austausch des Amsdos-ROM (der Sitz des Diskettenbetriebssystems im Controller) gegen ein EPROM vom Typ 27128. Die geringere Stromaufnahme dieses Bauteils sorgt für die nötige Entlastung und somit für einen fehlerfreien Betrieb. Voraussetzung für diesen Eingriff ist natürlich, daß man den Inhalt des ROMs mit Hilfe eines EPROM-Brenners in das EPROM kopiert.

Während die dk'tronics-Erweiterungen einfach anzustecken sind, kommt die Platine (siehe Bild 2) der Vortex-Speichererweiterung ins Computergehäuse. Man braucht für den Einbau ein wenig handwerkliches Geschick, was sicher nicht jedermanns Sache ist. Man muß sich immer der Konsequenz bewußt sein, daß die Garantie für den Computer erlischt. Auch entstehen im Gebrauch einiger Programme Probleme, die sich mit einem eigens zu diesem Zweck vorhandenen Basic-Befehl leider nicht in jedem Fall umgehen lassen. Wenn ein Programm wegen der eingebauten Speichererweiterung und trotz der Verwendung des Befehls DISBOS (desaktiviert das Betriebssystem der Erweiterung) nicht richtig arbeitet, ist man in der Regel machtlos.

Die Praxis zeigt, was Sache ist

Wie bewähren sich nun Speichererweiterungen im täglichen Einsatz? Das Arbeiten mit dem großen CP/M gestaltet sich sehr angenehm und macht einige Programme überhaupt erst zugänglich. Vor allem beim Vollausbau ist die 444 KByte große RAM-Disk bald nicht mehr wegzudenken. Man gewöhnt sich schnell an die immensen Vorteile im Gebrauch und ärgert sich bei fehlender RAM-Disk über den »langsamen« Diskettenzugriff. Leider steht die Vortex-Speichererweiterung nur den Besitzern eines CPC 464 oder 664 zur Wahl. Für den CPC 6128 gibt es bis heute nur die dk'tronics-Erweiterungen. Allerdings arbeitet der CPC 6128 damit geradezu ideal zusammen. Die Speichererweiterung bringt in Verbindung mit der Silicon Disc eine bis zu 444 KByte große RAM-Disk unter CP/M 2.2 und Plus. Aber bereits mit der 256 KByte großen Silicon Disc allein läßt sich hervorragend arbeiten.

Die 64-KByte-Erweiterung für die beiden kleineren Modelle CPC 464 und 664 macht am CPC 6128 keinen Sinn, denn seine zweite 64-KByte-RAM-Bank leistet denselben Dienst.

Es ist wirklich nicht einfach, allen Details der beiden Erweiterungen gerecht zu werden. Eine grobe Trennung bietet sich aber an. Besitzern eines CPC 6128 ist zweifelsohne mit der dk'tronics-Silicon-Disc und eventuell einer zusätzlichen 256-KByte-Erweiterung hervorragend gedient. Diese Erweiterungen sind ohnehin die einzig verwendbaren für diesen Computertyp, da Vortex noch kein Pendant anbietet.

Deutschen Besitzern eines CPC 464 oder 664 dürfte die Vortex-Erweiterung durch die große Verbreitung und die fertige Anpassung vieler Standard-Programme von größerem Reiz sein, wenn auch die Installationsprozeduren durch Umbauarbeiten erheblich komplizierter sind als bei dk'tronics. Geht es nur um das größere CP/M, ist die 64-KByte-Erweiterung von dk'tronics für 129 Mark ein preisgünstiger Griff. Alle anderen Kombinationen liegen preislich über dem Vortex-Komplettpreis. Wer seinen Computer optimal ausnutzen möchte, sollte diese Tatsachen gegeneinander abwägen.

(Helmut Jungkunz/gn/ja)

SP256/512: Vortex Computersysteme, Falterstraße 51-53, 7101 Flein
dk'tronics: Michael Naujoks, Rottmannstraße 40, 6900 Heidelberg

Ausstattung	Vortex SP 256 SP 512	dk'tronics Speicher-erweiterung	dk'tronics Silicon Disc
CPC 464	x	x (64/256 KByte)	(256 KByte)
CPC 664	x	x (64/256 KByte)	(256 KByte)
CPC 6128		x (256 KByte)	(64/256 KByte)
größere TPA unter CP/M 2.2	x	x	
CP/M Plus für CPC 464 und 664			
Drucker-Spooler	bis 512 KByte		
RAM-Disk unter CP/M CP/M und Basic	bis 444 KByte		bis 444 KByte
Bildsequenzen	x	x	
relativer Dateizugriff	x		
RSX-Befehlserweiterung	66 (resident)	12 (Software)	3 (resident)
größerer Basic-Variablen-speicher	x	x	
größerer Basic-Programm-speicher	x		
residenter Monitor	x		

Preise:		
Vortex SP 256 SP 512	(nur CPC 464/664)	298 Mark 398 Mark
dk'tronics Speichererweiterung 64 KByte 256 KByte 256 KByte	(464/664) (464/664) (6128)	129 Mark 298 Mark 298 Mark
Silicon Disc 64 KByte 256 KByte 256 KByte	(6128) (464/664) (6128)	98 Mark 298 Mark 298 Mark
Zu den CPC 6128-Versionen kommt noch ein Adapter zu 29 Mark hinzu		

Trickreiches Trio: Kopiermodule im Vergleich

Dem Herstellen von Sicherheitskopien bereiten ausgeklügelte Kopierschutzverfahren oft ein jähes Ende. Um diesem Umstand Abhilfe zu schaffen, gibt es neben Kopierprogrammen auch Hardware-Lösungen. Wir stellen Ihnen erstmals in einem Vergleichstest alle drei verfügbaren Kopiermodule vor.

Wer schon einmal die leidvolle Erfahrung machte, daß ein für teures Geld erworbenes Programm plötzlich auf dem Datenträger nicht mehr lesbar war, weiß Sicherheitskopien sehr zu schätzen. Um so ärgerlicher ist es, daß viele Softwarehersteller immer noch ihre Programme mit allerlei Kopierschutzmechanismen versehen. Ebensoviel praktischen Nutzen bringt die Übertragung von langsam ladender Kassettensoftware auf das schnellere Speichermedium Diskette.

Ein Kopiermodul, so die Versprechungen der Hersteller, räumt mit diesen Mißständen endgültig auf. Ein einfacher Knopfdruck soll den Benutzer in die glückliche Lage versetzen, alles zu kopieren, was in den Arbeitsspeicher seines Computers kommt. Ausgeschlossen davon sind allerdings bei allen Modulen Programme, die Grafiken oder sonstige Daten oder Programmteile nachladen.

Mirage Imager – der »Stammvater«

In der Tat ist der Imager das älteste und mit zirka 180 Mark auch eines der teuersten der getesteten Module. Sein Design paßt sich – am Expansionsport angeschlossen – sowohl farblich als auch vom Volumen her tadellos an das des CPC an. An seiner Oberseite fällt jedoch ein kleiner roter Knopf ins Auge. Übrigens gibt es, und das trifft auf jedes der getesteten Module zu, für den mit anderen Steckanschlüssen (Amphenolbuchsen) ausgestatteten CPC 6128 einen Adapter für zwischen knapp 30 und 40 Mark.

Einmal am CPC (mit oder ohne Laufwerk) angeschlossen, deutet zunächst gar nichts auf die Anwesenheit des Moduls hin.

Erst nachdem sämtliche Ladeprozeduren vollzogen sind und das zu kopierende Programm komplett im Arbeitsspeicher steht, folgt der Druck auf das rote Knöpfchen. Nun grüßt die zweizeilige Einschaltmeldung des Imagers. Darunter hat der Bildschirminhalt des öfteren jedoch vorübergehend recht heftig zu leiden. Ein anschließender Druck auf die ENTER-Taste bringt die Hauptmenüleiste auf den Bildschirm.

Das Menü bietet eine ganze Reihe Funktionen. Als wohl wichtigste betrachten wir zunächst die »Save«-Funktion. Nach Eingabe des Anfangsbuchstaben <S> wählt man zunächst den gewünschten Datenträger. Der Imager bietet die Auswahl zwischen den Diskettenlaufwerken A und B (sofern vorhanden) sowie dem Kassettenrecorder mit variabler Übertragungsrate bis 3000 Baud.

Nach anschließender Eingabe des Dateinamens startet der Speichervorgang: Der Bildschirm erlischt und es beginnt das äußerst langwierige Speichern, das bis zu zwei Minuten (Diskette) dauert.

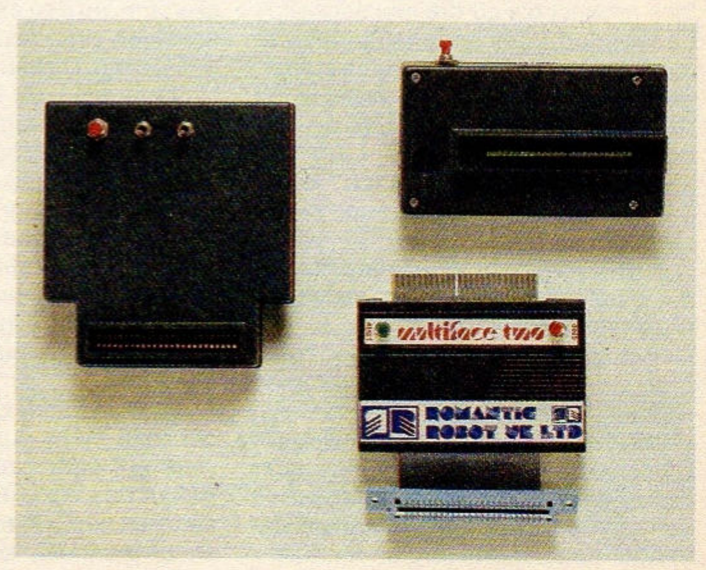
Mit der erneuten Anzeige des Hauptmenüs meldet sich der Computer danach wieder zur Stelle. Ein Druck auf die Taste <C> zeigt den Disketteninhalt an und offenbart, daß der Imager das Programm als bis zu 60 KByte lange Datei ablegt. Das liegt an der Arbeitsweise des Moduls. Es durchforstet den gesamten Arbeitsspeicher und berücksichtigt dabei auch die zweite 64-KByte-RAM-Bank des CPC 6128.

Eine vom Imager gespeicherte Datei läßt sich auch nur mit dem Modul wieder laden; die Kopien sind ohne den Imager nicht lauffähig. Damit ist sichergestellt, daß der Benutzer keine Raubkopien mit dem Imager anfertigen kann.

Um eine gespeicherte Programmkopie zu laden, drückt man wiederum den roten Knopf. Auch die weitere Vorgehensweise ähnelt dem Speichern, nur daß man <L> für »Load« drückt. Anschließend startet das kopierte Programm mit <R> genau an der Stelle, an der es zum Speichern unterbrochen wurde.

Der Imager erlaubt diverse Eingriffe

Nebenbei bietet der Imager noch die Wahl, die Bildschirmfarben zu wechseln, was besonders Besitzern eines Grünmonitors zustatten kommt. Auch das Format der Bildschirmdarstellung sowie – falls nötig – die Basisadressen des Bildschirmspeichers lassen sich beeinflussen. Sogar eine Abschaltung der externen ROMs ist vorgesehen. Das erweist sich mitunter vor allem bei sehr langen Kassettenprogrammen als nötig.



Alle getesteten Kopiermodule auf einen Blick: Rechts oben erkennen Sie den Mirage Imager, darunter das Multiface Two und auf der linken Seite den Disc Wizard

Die Dokumentation ist leider wie so oft vollständig in Englisch gehalten, läßt aber die wesentlichen Funktionsprinzipien klar erkennen.

Ein ungleicher Bruder

Äußerlich gibt sich das Multiface Two dem Mirage Imager gegenüber recht verschieden. Während der Imager sich förmlich an das Gehäuse des CPC schmiegt, ist das Multiface über ein Flachbandkabel mit dem Erweiterungsport verbunden und findet so hinter dem Computer liegend auf dem Tisch Platz. Im praktischen Einsatz jedoch offenbart sich eine nähere Verwandtschaft, als man zunächst vermutet. Ähnlich dem Imager zeigt auch das Multiface einen roten Knopf an seiner Oberseite. Daneben sitzt jedoch noch ein zusätzlicher grüner Resetknopf, den man am CPC oft vermißt.

Das Modul-eigene 8-KByte-RAM läßt sich vom Anwender für eigene Maschinencode-Routinen nutzen. Darüber weiß das zweiseitige, englischsprachige »Handbuch« jedoch nur wenig zu berichten.

Ist das Programm vollständig in den Speicher geladen und das Modul angeschlossen, aktiviert man es wie den Imager per Knopfdruck. Der Bildschirm gerät auch hier ein wenig in Unordnung, wenn am unteren Bildrand das Hauptmenü erscheint. Dort stehen vier Menüpunkte bereit. Wie beim Imager dient <S> dem Speichern. Auch die Auswahl der Speichermedien ist dem Konkurrenten ähnlich: Diskette oder Band mit bis zu 3000 Baud. Die Multiface-Kopien sind ohne das Modul ebenfalls nicht lauffähig.

Der Speichervorgang geht hier aber erheblich flotter vonstatten; das Multiface arbeitet bis zu dreimal schneller. Es legt den Speicherinhalt in mehreren Dateien ab. Das Programm startet, wenn man die erste dieser Dateien mit »RUN "NAME"« lädt. Genau wie beim Imager startet der Lauf des kopierten Programms durch <R> (für »RETURN«) wieder an der Stelle, an der er gestoppt wurde.

Zusätzlich spendierten die Entwickler dem Multiface einen einfachen Maschinencode-Monitor. So lassen sich einzelne Register lesen und beliebige Adressen in den Programmen ändern. Spielefreaks können sich damit beispielsweise ihre individuelle Spieleversion »zurechtbasteln« oder Spielfiguren zum unendlichen Leben verdammen. Für zirka 180 Mark ist der Imager eines der teuersten Module.

Disc Wizard unterscheidet sich in vielerlei Hinsicht von seinen beiden Artgenossen. So ist es das einzige Modul, zu dessen Lieferumfang eine Diskette gehört.

Das äußere Erscheinungsbild unseres Dritten im Bunde ist ebenfalls ungewöhnlich. Am Expansions-Port angeschlossen, ragt das Modul etwa 10 cm hoch über die Tastatur. An der Vorderseite fanden neben dem obligatorischen roten Knöpfchen zwei Kippschalter Platz. Die Schalterstellungen sind mit »Load« und »Save«, beziehungsweise »1« und »2« gekennzeichnet. Damit sind wir beim zweiten wesentlichen Unterscheidungsmerkmal des Wizard zu seinen Konkurrenten. Pro Diskettenseite bietet es nur Platz für zwei Programm-Kopien. Disc Wizard unterteilt nämlich den Speicherplatz einer Diskette im Data-Format in zwei getrennte Bereiche für je ein Programm. Leider darf man auf diesen präparierten Diskettenseiten keine zusätzlichen normalen Dateien speichern. Diesem eigensinnigen Verhalten steht jedoch eine wirklich kinderleichte Handhabung des Wizard gegenüber. Zum Kopieren des Programms im Speicher ist nur der linke Kippschalter auf »Save« und der rechte auf »1« oder »2« (zur Wahl eben jener Bereiche) zu bewegen und anschließend der Knopf zu drücken.

Utilities erleichtern die Arbeit

Ladevorgänge gestalten sich genauso unkompliziert. Diesmal steht der erste Schalter beim Knopfdruck auf »Load«. Damit entfällt bei diesen Prozeduren jede Eingabe via Tastatur. Das Modul benennt die Dateien automatisch mit den Namen »DRAYSOFT.ONE« und »DRAYSOFT.TWO«. Damit die Kopien verschiedener Programme auf den ersten Blick zu unterscheiden sind, findet sich auf der mitgelieferten Diskette eine Utility zum Umbenennen der Dateien. Aber auch andere Dienstprogramme sind darauf enthalten. So lassen sich sogenannte »Screenshots«, also Bildschirminhalte (Titelbilder etc.) im normalen Dateiformat erzeugen, aber auch Farben, Bildschirmmodi und Startadressen ändern. Die Hilfsprogramme erlauben damit ähnliche Eingriffe wie die anderen Module es aus ihren Menüs heraus tun.

Beim Studium des Handbuchs findet man mehr durch Zufall den Hinweis auf eine ganz bedeutsame Eigenart des Wizard. Einige Kopien lassen sich nämlich mit einer der Utilities so umwandeln, daß sie auch ohne Modul lauffähig sind. Eine Tatsache, die Softwarehersteller sicherlich nicht gerade zum Jubeln veranlassen wird. Der Disc Wizard kostet inklusive der Diskette etwa 150 Mark.

Ihre Aufgabe erfüllen alle drei Testkandidaten zur vollen Zufriedenheit. Und doch lassen sich einige deutliche Unterschiede in der Handhabung ausmachen. So fallen beim Imager die unnötig langen Speicher- und Ladezeiten negativ ins Gewicht. Verglichen mit ihm sind das Multiface Two und der Disc Wizard wahre »Turbo«-Module. Als besonders positiv ist jedoch die Konstruktion seines Gehäuses zu werten, das sich nahtlos an die Rückseite des CPC anfügt. Damit empfiehlt sich der Mirage Imager besonders Benutzern mit großer Geduld und kleinem Schreibtisch, zumal man mit dem Multiface oder dem Wizard vergleichbare Leistung zum gleichen oder gar niedrigeren Preis erhält.

Dreigestirn im Vergleich

Der leider nur mit Diskette erhältliche Disc Wizard beispielsweise erzeugt, wie schon erwähnt, Programmkopien, von denen nach Aufbereitung 70 Prozent auch ohne Modul lauffähig sind. Damit wartet der Wizard mit einem Ausstattungsmerkmal auf, das keine der beiden anderen Erweiterungen zu bieten hat. Einzigartig ist auch sein Bedienungskomfort. Zu seinen unangenehmeren Eigenschaften zählt sein spezielles Aufzeichnungsformat, das nur vier Kopien pro Diskette zuläßt.

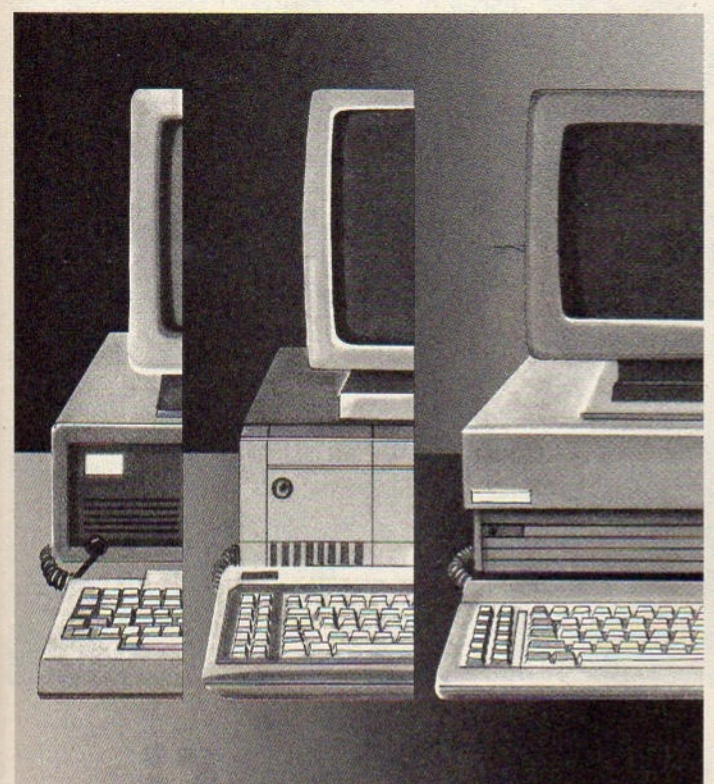
Ganz anders präsentiert sich da das Multiface Two, das sowohl schnell arbeitet, als auch keine eigens vorbereiteten Disketten verlangt. Sein eingebauter Resetknopf und der für Maschinensprache-Programmierer interessante Speichermonitor machen ihn zur lohnenswerten Anschaffung.

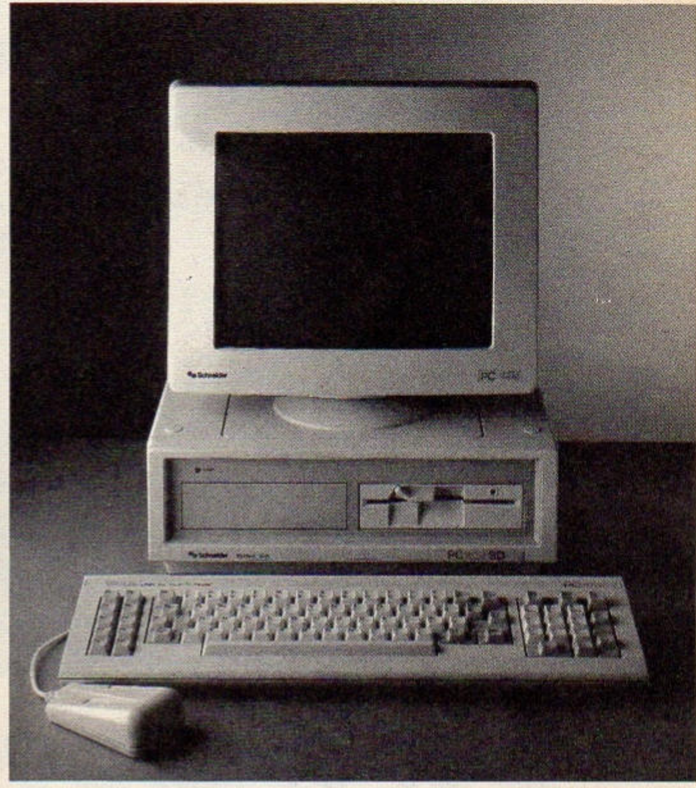
Eine Warnung sei jedoch an dieser Stelle ausgesprochen: Jedes der drei Module reagiert empfindlich auf die Anwesenheit fast jedweder Fremdperipherie am CPC. Unter anderem läßt sich nämlich keines auf eine friedliche Koexistenz mit Vortex-Laufwerken ein. Wir können Ihnen daher nur raten, vor dem Kauf eines der Module die Funktion mit Ihrer Systemkonfiguration ausgiebig zu testen.

(Ralf Hinnenberg/kl/ja)

Mirage Imager: Mirage Microcomputers Deutschland, Postfach 160155, 5400 Koblenz 16
Multiface Two: Romantic Robot, Ben-Gurion-Ring 80, 6000 Frankfurt am Main 56
Disc Wizard: Waldeck Software, Tulpenstraße 30, 2870 Delmenhorst

Wieviel Computer können Sie für 1499,- kaufen?





ca. soviel von Vielen | soviel von Schneider

Ab DM 1.499,–* erhalten Sie einen IBM**-kompatiblen Komplett-Computer: Monitor, Zentraleinheit, 5¼" Diskettenlaufwerk, Tastatur und …

- ★ 16 Bit-Prozessor 8086 mit 8 MHz Taktfrequenz
- ★ 512 Kilobyte Hauptspeicher
- ★ Betriebssysteme MS DOS 3.2 und DOS PLUS
- ★ grafische Benutzeroberfläche GEM
- ★ Microsoft-kompatible Maus
- ★ eingebaute Schnittstellen für Peripherie und Datenfernübertragung
- ★ Auflösung 720 × 348 Bildpunkte monochrom (Option)

Vielseitig, bedienungsfreundlich und sagenhaft preiswert.

Der neue Schneider »PC 1512«.

Mit Industriestandard, GEM und Maus bietet er komplett, was bisher nur einzeln zu haben war:

Die IBM-Kompatibilität für den Zugriff zur größten Softwarebibliothek der Welt. Die Schnelligkeit für professionelle Anwendung. Die GEM-Benutzeroberfläche, die auch den Anfänger ohne großes Computerwissen im Handumdrehen zum Computerprofi macht.

Die Zukunftssicherheit durch nahezu unbegrenzte Ausbau- und Anschlußmöglichkeiten (Modems, Netzwerke, vielfältige Peripherie).

Und den Preis, der für eine erfolgreiche Zukunft völlig neue Perspektiven eröffnet.

Die Produktfamilie:

Modell	Unverbindliche Preisempfehlung
Schwarz-/Weiß-Monitor, ein Diskettenlaufwerk	1.499,- DM
Schwarz-/Weiß-Monitor, zwei Diskettenlaufwerke	1.999,- DM
Schwarz-/Weiß-Monitor, ein Diskettenlaufwerk, 20 MB Magnetplattenlaufwerk	2.999,- DM
Farbmonitor, ein Diskettenlaufwerk	1.999,- DM
Farbmonitor, zwei Diskettenlaufwerke	2.499,- DM
Farbmonitor, ein Diskettenlaufwerk, 20 MB Magnetplattenlaufwerk	3.499,- DM

Coupon

Bitte schicken Sie mir Ihre komplette Information über den Schneider »PC 1512«.
Meine Anschrift:

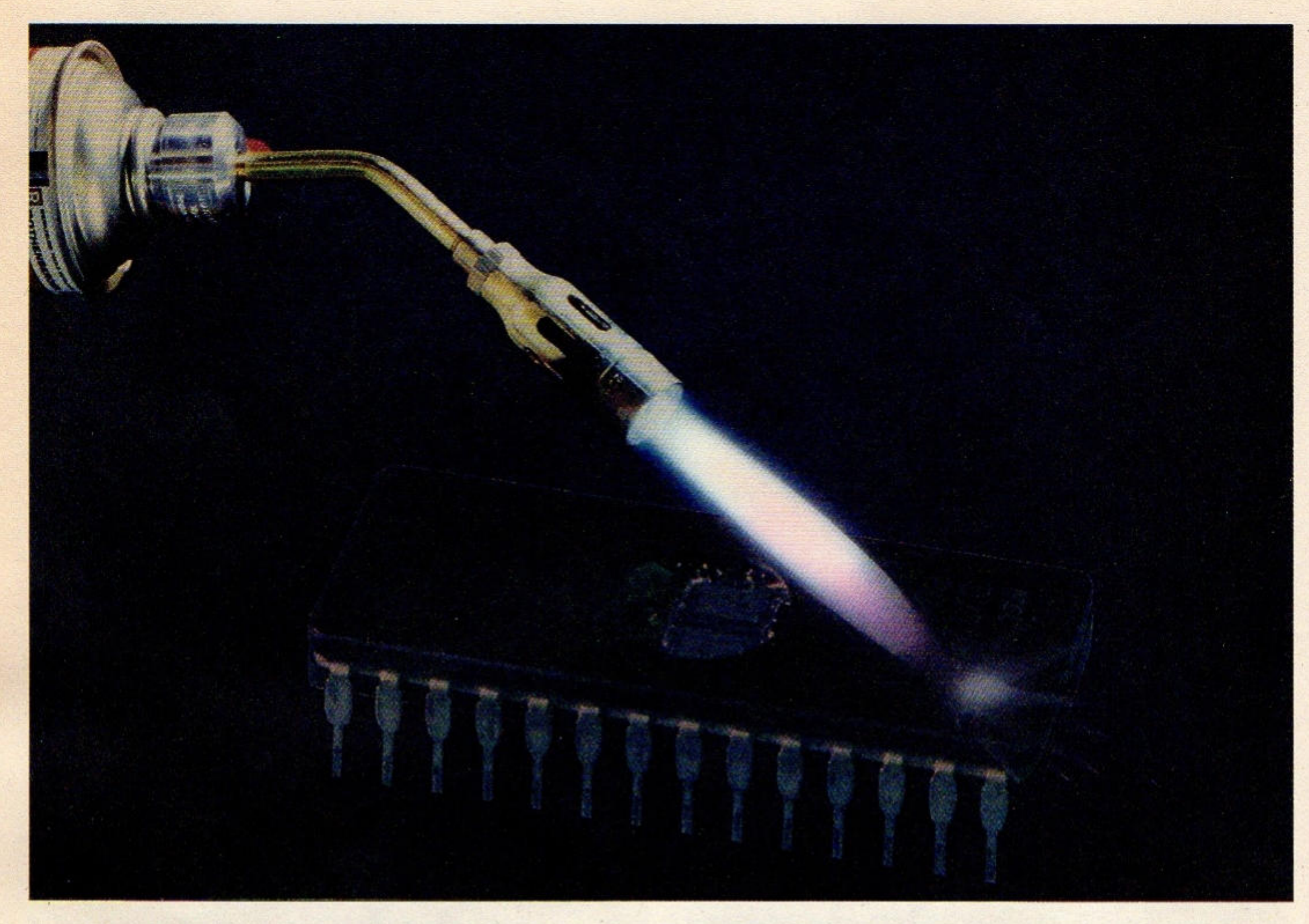
Name

Straße/Nr.

PLZ/Ort

Schneider Info-Service, Widenmayerstr. 34, 8000 München 22

* Unverbindliche Preisempfehlung
** IBM ist ein eingetragenes Warenzeichen von International Business Machines Corp.



Der Happy-EPROMer: ein Tausendsassa

Jetzt ist es soweit – der Happy-EPROMer für den CPC ist da. Einfach nachbauen, aufstecken sowie die zugehörige Software starten, und schon sind Sie in der Lage, eigene Programme in EPROMs zu brennen und beispielsweise das Betriebssystem des CPC Ihren eigenen Wünschen anzupassen.

Viel Mühe und Schweiß hat uns die Entwicklung des Happy-EPROMers gekostet, doch das Ergebnis kann sich sehen lassen! Sie erhalten mit dem Happy-EPROMer eine komplett softwaregesteuerte Schaltung, die EPROMs mit einer Speicherkapazität von 8 bis 32 KByte über einen intelligenten Programmier-Algorithmus programmiert und direkt am CPC aufgesteckt werden kann (Bild 1).

Sie können sowohl die normalen EPROMs benutzen, die mit einer Spannung von 21 Volt programmiert werden, als auch die A-Typen, die nur 12,5 Volt Programmierspannung benötigen.

Der Clou der Schaltung ist, daß die Spannung intern mit einem Spannungswandler erzeugt wird, so daß keine externe Spannungsquelle notwendig ist.

Selbstverständlich liefern wir Ihnen auch die passende Software zum Happy-EPROMer, denn ohne Software ist die schönste Schaltung wertlos.

Mit dem Happy-EPROMer eröffnet sich für Sie ein weites Feld an Anwendungsgebieten. Von der residenten RSX-Befehlserweiterung im EPROM über die selbstgeschriebene oder gekaufte Software, die nach Einschalten des Gerätes sofort verfügbar ist, bis hin zur gepatchten Version des CPC-Betriebssystems ist nun alles machbar.

Besonders günstig ist, daß das Betriebssystem des Schneider CPC für den Anschluß von bis zu 252 EPROMs vorbereitet ist. So lassen sich die programmierten EPROMs beispielsweise über eine ROM-Box (siehe auch Testbericht in dieser Ausgabe) oder über die im 7. Schneider-Sonderheft vorgestellte Happy-Megabitkarte anschließen und komfortabel in das Betriebssystem einbinden.

Wenden wir uns nun dem konkreten Aufbau des Happy-EPROMers zu. Bild 2 zeigt seine wichtigsten Komponenten, die aus nur vier ICs bestehen. Lediglich der Schaltungsteil zur Erzeugung der Programmierspannung und der Textool-Sockel, in den die EPROMs gesteckt werden, sind aus Gründen der Übersichtlichkeit nicht eingezeichnet.

Wenn man die EPROMer-Schaltung betrachtet, fällt zuerst das IC 8255 ins Auge. Bei diesem Baustein handelt es sich um ein IC mit drei 8-Bit-

Ports zur parallelen Datenein- und -ausgabe (PIO).

Dieser Baustein ist das Herz des Happy-EPROMers. Hier werden Daten aus dem CPC an das EPROM übergeben und Daten vom EPROM in den CPC übernommen.

Über ein 0-Signal auf den Adreßleitungen A8 bis A10 adressiert der CPC den 8255 (Portadresse F8XX), und über die beiden Adreßbits A1 und A2 wählt er einen Port (A, B und C) oder das Steuerregister zur Wahl der Betriebsart des 8255 aus.

Die Signale $\overline{RD}$ und $\overline{IORQ}$ sowie $\overline{WR}$ und $\overline{IORQ}$ werden jeweils über ein ODER-Gatter miteinander verknüpft und an den $\overline{RD}$-Eingang beziehungsweise $\overline{WR}$-Eingang des 8255 gelegt. Diese beiden Signale geben an, ob Daten in den 8255 geschrieben oder aus dem Baustein gelesen werden.

Der Datenaustausch zwischen CPC und 8255 erfolgt über den Datenbus (D0 bis D7). Weil die Signale des Datenbus nicht durch einen Treiber (zum Beispiel 74LS245) verstärkt werden, müssen diese Leitungen so kurz wie möglich gehalten werden. Der 8255 sollte folglich auf der Platine direkt am Stecker zum CPC sitzen. Im anderen Fall müssen Sie noch ein IC 74LS245 zur Verstärkung der Datensignale einfügen.

Drei Ports im Streß

Port A des 8255 gibt die zu schreibenden Daten an das EPROM aus und liest bei Bedarf die Daten vom EPROM ein.

Port B erzeugt die Signale zur Adressierung der Speicherzellen des EPROM. Zuerst werden die Adreßbits A8 bis A13 ausgegeben und in dem 8-Bit-Register 74HC573 zwischengespeichert. Sofern das EPROM gelesen werden soll, wird zusätzlich das $\overline{OE}$-Signal aktiviert.

Darauf wird über Port B das untere Byte der EPROM-Adresse, nämlich die Adreßbits A0 bis A7 ausgegeben und an den Textool-Sockel angelegt. (Die Zahlen in Klammern hinter den Signalen auf der rechten Seite des 8255 geben jeweils die Nummer des angeschlossenen Pin am Textool-Sockel an.)

Port C schließlich erzeugt die Steuersignale für die EPROMer-Schaltung. Bit 0 aktiviert das 8-Bit-Register, um das obere Adreßbyte von Port B zu übernehmen, und Bit 1 schaltet das EPROM über die $\overline{CE}$-Leitung in den aktiven Zustand.

Bit 2 schaltet die Programmierspannung für das EPROM ein, und Bit 6 wählt aus, ob die Programmierspannung 21 Volt oder 12,5 Volt beträgt. Beide Signale sind mit einem Inverter aus dem IC 74LS06 gepuffert. Die Ausgänge der Gatter dieses IC verfügen über offene Kollektoren, so daß die Ausgangsspannung eines Gatters bei entsprechender Beschaltung bis zu 30 Volt betragen darf. Auf diese Weise lassen sich bequem die TTL-Signale der digitalen Elektronik (+5 Volt und 0 Volt) mit den Spannungen zum Programmieren eines EPROM koppeln.

Bit 7 von Port C erzeugt das Programmiersignal $\overline{PGM}$ für das EPROM und wählt beim EPROM-Typ 27256 als Adreßbit A14 die untere oder obere Speicherhälfte aus.

Bit 3, Bit 4 und Bit 5 von Port C werden in der Schaltung nicht verwendet und bleiben unbeschaltet.

Der Schaltungsteil des Happy-EPROMers, der die Programmierspannungen für das EPROM erzeugt, besteht aus dem Spannungswandler TL 497 von Texas Instruments und seiner Beschaltung, die aus nur wenigen Bauteilen besteht.

Der TL 497 ist in der Lage, aus einer bestimmten Eingangsspannung Ausgangsspannungen von bis zu 35 Volt bei einer maximalen Stromstärke von 0,5 Ampere zu erzeugen. Mit der angegebenen Beschaltung erzeugt er aus den 5 Volt, die der CPC am Erweiterungsanschluß zur Verfügung stellt, eine Spannung, die je nach Einstellung des Potentiometers R1 zwischen 20 und 32 Volt liegt.

Für dieses kleine Wunder benötigt das IC jedoch die Unterstützung durch eine Spule, die Sie selbst herstellen müssen, sofern Sie im Fachhandel nicht eine passende Spule mit dem Al-Wert von etwa 250 finden.

Doch keine Angst, der Selbstbau der Spule ist völlig unproblematisch. Aus einem Elektronikladen besorgen Sie sich dazu einen Spulenbausatz, bestehend aus zwei Kammerhälften Ferritschalen, einem Spulensockel mit Wickelzylinder und einem Klammerbügel. Empfehlenswert sind die Spulenbausätze von Valvo oder Siemens. Der Al-Wert des Ferritmaterials muß ungefähr 250 (zum Beispiel Siemens-Schalenkern RM 5, N28) betragen.

Wie die Erfahrung zeigt, sind diese Werte jedoch relativ unkritisch. Den dem Bausatz beiliegenden Wickelzylinder bewickeln Sie bitte mit 15 Windungen Kupferlackdraht, der eine Drahtstärke von 0,3 bis 0,4 Millimeter besitzen sollte und setzen darauf die Spule gemäß beiliegender Anleitung zusammen.

Für die Qualität der Ausgangsspannung ist es hilfreich, wenn Sie die Beschaltung des TL 497, wie im Verdrahtungsplan auf Bild 3 zu sehen, so nahe wie möglich am Sockel des IC aufbauen. Dadurch werden Rückkopplungen vermieden.

Es ist sinnvoll, den Programmierspannung erzeugenden Teil der Schaltung zuerst aufzubauen und eine externe Spannungsquelle (zum Beispiel eine 9-Volt-Blockbatterie) anzuschließen. So können Sie mit dem Potentiometer R1 die Ausgangsspannung der Schaltung auf 22 Volt voreinstellen.

Der Programmierspannung auf der Spur

Die Programmierspannung, die vom TL 497 erzeugt wird, liegt direkt an der Zenerdiode D1 an, die eine Zenerspannung von 22 Volt aufweist, und bei einer Fehlfunktion des TL 497 oder der Beschaltung (Kurzschluß im IC, defekter Kondensator oder verdrehtes Potentiometer) dafür sorgt, daß Überspannungen kurzgeschlossen werden. Dadurch besteht keine Gefahr, daß ein EPROM durch zu hohe Spannungen zerstört wird.

In dieser Richtung finden Sie im Verdrahtungsplan auch die beiden Transistoren T1 und T2, eine weitere Zenerdiode D2, eine Germaniumdiode D3 und eine Siliziumdiode D4. Diese Bauteile erzeugen die beiden Programmierspannungen von 21 und 12,5 Volt für Pin 1 des Textool-Sockels.

Die Widerstände dienen lediglich zur Strombegrenzung, doch ein

Bild 1. So läßt sich der Happy-EPROMer ohne Aufwand am CPC anschließen

Amiga-Magazin, das Computer-Magazin für Amiga-Fans, die Zeitschrift für alle Commodore-Amiga-Besitzer

- hilft Ihnen, den Amiga maximal zu nutzen
- bringt für Einsteiger und Experten, Hobby- und Profiprogrammierer-Kurse in CLI, BASIC, MODULA II, »C« etc. etc.
- testet für Sie Hardware, Peripherie und aktuellste Software sämtlicher Hersteller
- anspruchsvolle Listings und Anwendungen geben Ihrer Arbeit höchste Effizienz
- in Kursen optimieren Sie die Bedienung Ihres Amiga

Das Amiga-Magazin kommt am 27. Mai '87

1/87

AMIGA MAGAZIN

DAS COMPUTERMAGAZIN FÜR AMIGA-FANS

Ausführliche Tests

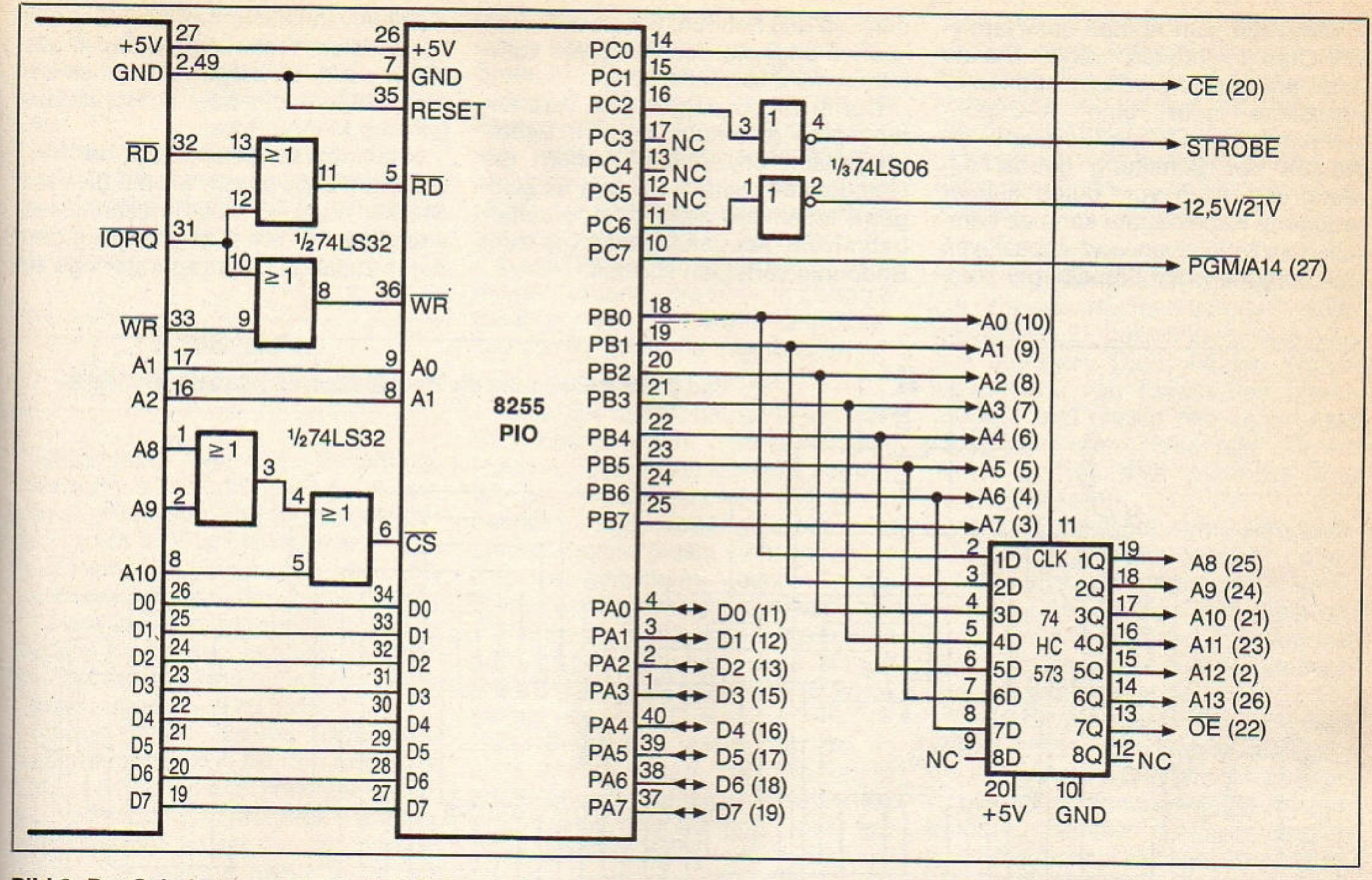


Bild 2. Der Schaltplan zeigt die digitale Logik des EPROMers und die Anschlüsse für den Textool-Sockel

0-Volt-Impuls an der Basis von T1 (STROBE) schaltet die Transistorstrecke zwischen Emitter und Kollektor frei und damit die um ein knappes halbes Volt reduzierte Programmierspannung über T2 oder D2 an das EPROM.

Liegt STROBE dagegen auf 1, so ist der Transistor T1 gesperrt, und über die Diode D3 liegen knapp 5 Volt an Pin 1 des EPROM, so daß es sich nicht programmieren läßt.

Das Signal an der Basis von T2 (12,5/21) kann die Programmierspannung von 21 auf 12,5 Volt begrenzen, indem es den Transistor gesperrt hält. In diesem Fall muß sich die Programmierspannung ihren Weg über die Zenerdiode D2 suchen, die den Spannungswert um ungefähr 9 Volt reduziert.

Ist T2 jedoch durchgeschaltet, so wird die Programmierspannung über die Emitter-Kollektor-Strecke des Transistors zum EPROM geleitet und verliert wiederum nur ein knappes halbes Volt an Spannung. Dadurch liegen ziemlich genau 21 Volt an Pin 1 des Textool-Sockels, sofern der Transistor T1 durchgeschaltet ist.

Viele Leser werden sich fragen, wozu der Aufwand mit der Programmierspannung notwendig ist und warum man nicht die Betriebsspannung von 5 Volt, die der CPC serienmäßig liefert, verwenden kann. Die Antwort ist kurz, die Begründung dafür etwas länger: Der interne Aufbau eines EPROM erfordert diese relativ hohen Spannungen, um eine Speicherzelle dauerhaft (für mindestens 10 Jahre) zu programmieren.

Ein EPROM enthält im Inneren viele tausend kleine MOSFET-Transistoren, die sperrend geschaltet sind, folglich den Transport von Ladungsträgern verhindern. Normale Transistoren lassen sich über eine Spannung an der Basis freischalten, im EPROM ist dies jedoch nicht möglich.

Wie ein EPROM funktioniert

Das im EPROM für den Transistor verwendete Material leitet den Strom schlecht und wird in der Elektrotechnik als Isolator bezeichnet. Nur mit einer zur Betriebsspannung von +5 Volt relativ hohen Spannung läßt sich ein von der Zenerdiode bekannter Lawineneffekt auslösen, durch den Ladungsträger über den MOSFET-Transistor transportiert werden.

Nach Abschalten der Programmierspannung bleiben die verschobenen Ladungsträger dann auf der gegenüberliegenden Seite des Transistors eingesperrt. Wenn Sie diesen programmierten Zustand des EPROM rückgängig machen wollen, müssen Sie den umgekehrten Weg gehen, das heißt eine hohe Energie einbringen, die den Rückfluß der eingesperrten Ladungsträger über den Transistor veranlaßt. Dieser Vorgang, der dem Löschen einer Speicherzelle entspricht, wird nicht durch das Anlegen einer hohen Programmierspannung, sondern durch die Einwirkung von UV-Strahlen auf den Baustein ausgelöst. Die UV-Strahlen setzen einen fotoelektrischen Strom in Gang, der die Ladungsträger in ihre Ausgangsstellung zurückfließen läßt.

Beim Löschen eines EPROMs läßt sich natürlich nicht eine einzelne Speicherzelle adressieren. Deshalb kann man mit einer UV-Lampe jeweils nur den kompletten Speicherinhalt eines EPROMs löschen.

Damit die UV-Strahlen auch die MOSFET-Transistoren des EPROM beleuchten können, bauen die EPROM-Hersteller in jeden Baustein ein kleines Fenster aus Quarzglas ein. (Quarzglas deshalb, weil es im Gegensatz zum normalem Fensterglas den UV-Anteil des Lichts nicht ausfiltert.) Dieses Fenster sollte bei einem programmierten EPROM abgedeckt werden, damit der Speicherinhalt durch die kontinuierliche Einwirkung des Tageslichts nicht gelöscht wird.

Wenn Sie zum Aufbau des Happy-EPROMers schreiten, wählen Sie als Basis am besten eine Lochstreifenrasterplatine oder eine IC-Experimentierplatine. Gehen Sie beim Verdrahten der Schaltung getrost ein wenig chaotisch vor; durch sauber gezogene Kabelbäume kann es nämlich zum induktiven und kapazitiven Übersprechen der Signalpegel kommen, so daß Fehlfunktionen auftreten können oder die Schaltung von vornherein nicht korrekt arbeitet.

Der in Bild 3 dargestellte Verdrahtungsplan gibt auch den im Schaltungslesen ungeübten Bastlern die Gelegenheit, einmal »richtig loszulegen«, indem sie deutlich jede Leiterbahn vom Ausgangspunkt bis zum Endpunkt verfolgen können.

Selbstverständlich sollten Sie, wie bei unserem Testmuster in Bild 4, alle ICs sockeln. So ersparen Sie sich bei einer Fehlersuche oder im Reparaturfall eine Menge Arbeit.

Nochmals an dieser Stelle der Hinweis, den 8255 so nah wie möglich am Stecker zum Erweiterungsanschluß einzulöten, um Signalstörungen durch zu lange Übertragungswege zu

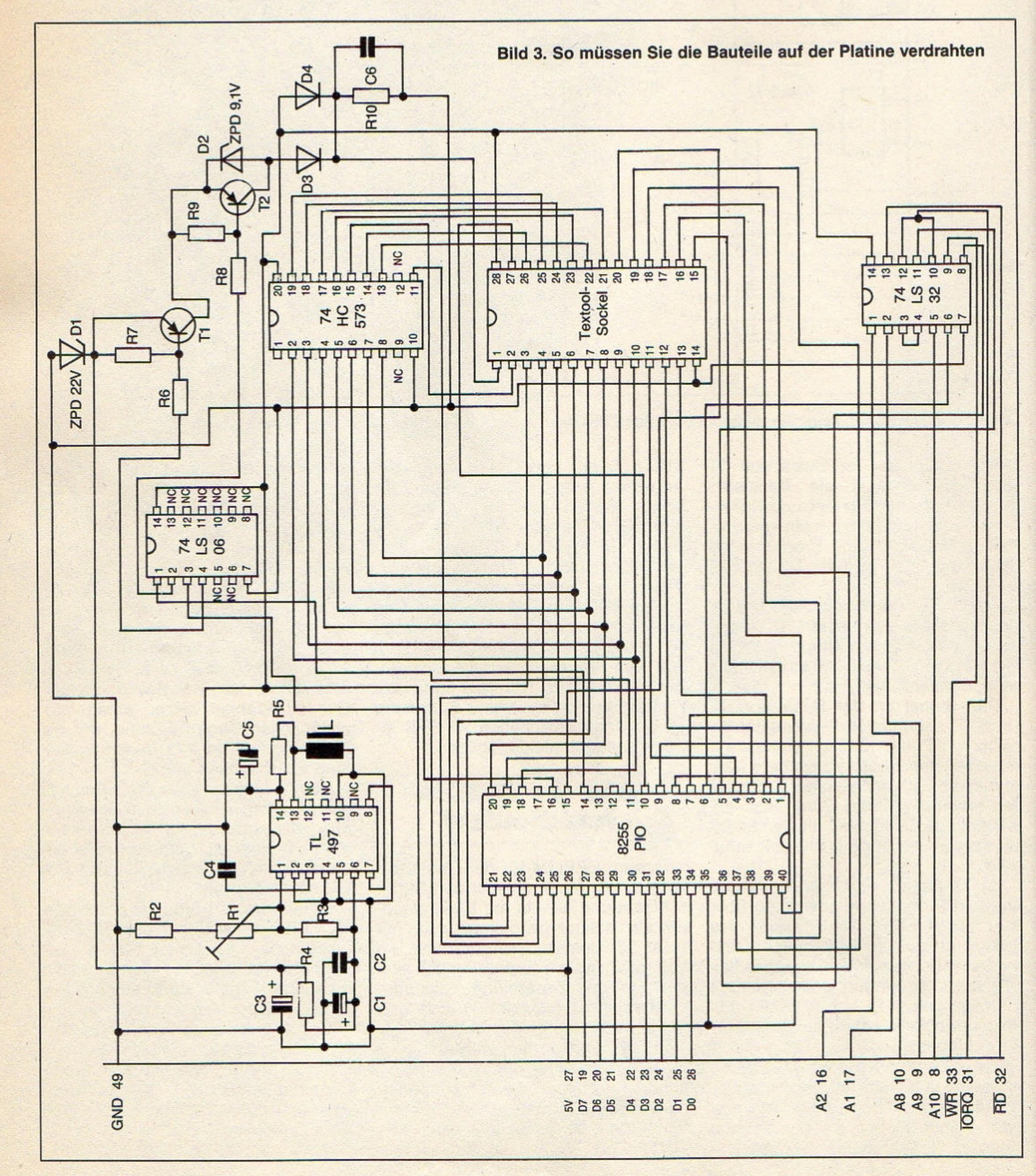


Bild 3. So müssen Sie die Bauteile auf der Platine verdrahten

vermeiden. Die Leitungen für die Portsignale des 8255 dürfen dagegen etwas länger ausfallen, weil diese Signale vom 8255 intern verstärkt werden.

Achten Sie bitte ebenfalls darauf, die Programmierspannung erzeugenden Bauteile dicht am IC TL 497 einzubauen, um störende Rückkopplungen zu vermeiden.

Tips aus der Bastelküche

Verdrahten Sie die Schaltung, indem Sie sich vorher eine Kopie von Bild 3 anfertigen und jeden gezogenen Draht im Plan farbig markieren. Beginnen Sie, nachdem Sie alle Sockel eingelötet haben, immer an Pin 1 eines Sockels und arbeiten Sie sich bis zum Pin mit der höchsten Nummer vor. So haben Sie die Gewähr, keine Leitung übersehen zu haben.

Auf die Platine unseres Testmusters ist ein Direktstecker für den Anschluß an den CPC und ein Direktstecker für die Verbindung zu weiteren Peripheriegeräten aufgelötet. Welche Kombination Sie wählen, bleibt selbstverständlich Ihnen überlassen.

Wenn Sie die Schaltung aufgebaut haben, können Sie den Happy-EPROMer im ausgeschalteten (!) Zustand des Computers auf den Erweiterungsanschluß stecken. Beim Einschalten des CPC muß sich die gewohnte Einschaltmeldung zeigen. Wenn nicht, dann schalten Sie den Computer schleunigst aus und begeben Sie sich auf Fehlersuche. Ist ein IC mehr als handwarm geworden? Sind alle Verbindungen ordentlich verlötet (notfalls mit einem Durchgangsprüfer nachmessen)? Haben Sie den Stecker richtig herum aufgesetzt?

Wenn Ihr Computer mit angeschlossenem EPROMer einwandfrei arbeitet, können Sie sich der Software zuwenden, die zum weiteren Überprüfen der Schaltung und selbstverständlich zum Programmieren eines EPROM unerläßlich ist.

Die Software für den Happy-EPROMer besteht einerseits aus dem Programm »Eprom«, das die EPROMer-Schaltung verwaltet, indem es die verschiedenen Speicherbausteine programmiert, ausliest und vergleicht, und andererseits aus einer angepaßten Version des CPC-Monitors »Smon«, der in der ersten Version bereits im 2. Schneider-Sonderheft abgedruckt wurde.

Beide Programme können sich gegenseitig aufrufen, so daß sie sich wie ein einziges Programm verhalten.

Bei Eprom geschieht der Aufruf über einen eigenen Menüpunkt und beim Smon mit dem Befehl »J 9000«.

Aus Platzgründen haben wir die Programme Eprom und Smon diesmal nicht als DATA-Lader, sondern abtippfertig für die Eingabehilfe »CPC« (siehe entsprechender Beitrag) abgedruckt (Listing 1 und Listing 2).

Gestartet werden beide Programme mit der Routine »Promon« in Listing 3. Besitzer des CPC 464 müssen Zeile 160 von Promon wie folgt ändern:

```
160 LOAD"smon.bin",&8000
```

Wenn Sie das Modell CPC 464 oder CPC 664 besitzen, müssen Sie die Routine »Patch« aus Listing 4 starten, die den Maschinencode von Smon nachlädt und abhängig von der Modellversion patcht. Anschließend wird der geänderte Maschinencode wieder auf Datenträger geschrieben.

Haben Sie die Software startbereit auf Diskette vorliegen, dann starten Sie sie nur noch mit »RUN"Eprom« gefolgt von <ENTER>.

Zur Bedienung von Eprom gibt es nicht viel zu sagen. Das Programm arbeitet mit einem intelligenten Programmier-Algorithmus, der ein 16-KByte-EPROM in weniger als zwei Minuten komplett programmiert.

Im Hauptmenü können Sie zwischen den verschiedenen EPROM-Typen wählen. Wenn Sie einen 27128 für 21 Volt Programmierspannung verwenden, müssen Sie das Untermenü für den 27128, ansonsten (12,5 Volt) das Untermenü für den 27128A auswählen. Das gleiche gilt für die beiden 27256-Typen.

Hüten Sie sich davor, beim Programmieren eines A-Typs im Menü den Typ ohne A auszuwählen. In diesem Fall wird das EPROM durch die zu hohe Programmierspannung zerstört. Damit Sie sich jederzeit über den gewählten EPROM-Typ informieren können, zeigt jedes Untermenü den aktuellen Typ in der Kopfzeile an. Haben Sie sich vertippt, so betätigen Sie die ENTER-Taste ohne Ihre Eingabe zu vervollständigen. In diesem Fall gelangen Sie in die Menüauswahl zurück.

Zum Austesten der EPROMer-Schaltung überprüfen Sie zuerst, ob die Programmierspannungen auch tatsächlich 21 beziehungsweise 12,5 Volt betragen. Dazu wählen Sie im Hauptmenü von Eprom den Menüpunkt 1 und darauf den Unterpunkt »Programmieren« (ebenfalls 1). Im Textool-Sockel darf natürlich kein EPROM stecken.

Auf die Frage nach dem zu programmierenden Speicherbereich geben Sie als Anfangsadresse 4000 (hex) und als Endadresse 7FFF (hex) an, damit Sie während des Programmierens genügend Zeit zum Messen der Programmierspannung haben.

Nach Eingabe der EPROM-Adresse (0000) startet der Happy-EPROMer die Programmierung und Sie müssen mit einem Spannungsmeßgerät oder einem Oszilloskop die Spannung an Pin 1 des Textool-Sockels überprüfen. Da die Programmierspannung in Form von Impulsen angelegt wird, ist ein analoges Spannungsmeßgerät wegen der Trägheit der Anzeige zum Messen besser geeignet als ein Digitalvoltmeter. Bei einem Digitalvoltmeter kann man sich notfalls mit dem Meßbereich Wechselspannung behelfen.

Stellen Sie das Potentiometer R1 mit einem kleinen Schraubenzieher so ein, daß die gemessene Spannung etwa 21 Volt beträgt.

Zum Kontrollieren der Programmierspannung von 12,5 Volt für die A-Typen wählen Sie im Hauptmenü von

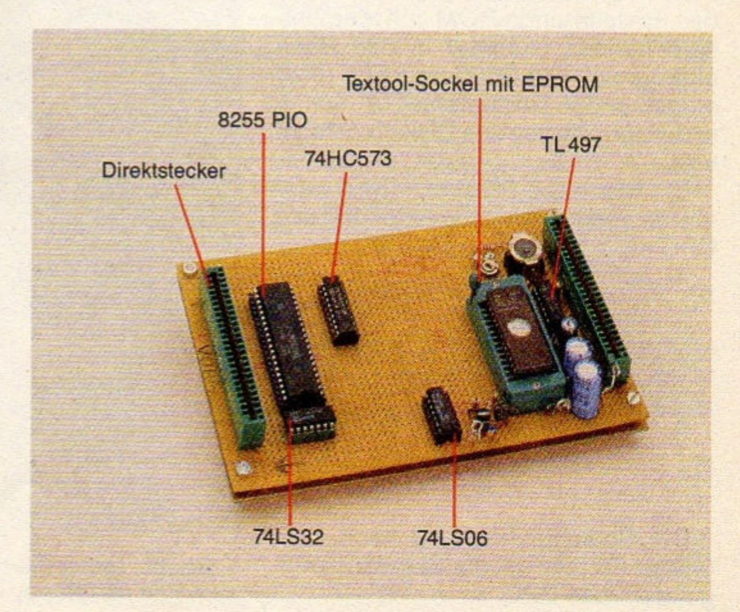


Bild 4. Ein Muster des Happy-EPROMers, das sich in der Redaktion bereits viele Male bewährt hat

Eprom den Punkt 4 aus. Nun können Sie einerseits an Pin 1 die Programmierspannung überprüfen und andererseits an Pin 27 testen, ob das Adreßsignal A 14 für die Auswahl der unteren und oberen Speicherhälfte eines 27256-EPROM korrekt erzeugt wird. Programmieren Sie das EPROM dazu einmal ab EPROM-Adresse 0000 und im anderen Fall ab EPROM-Adresse 4000 (hex). Im ersten Fall muß A14 auf 0 Volt und im zweiten Fall auf +5 Volt liegen.

Für die Verwaltung des CPC-Speichers (ROM, RAM und Erweiterungs-ROM) sowie für das Laden, Speichern und Ausführen von Programmen ist der CPC-Monitor Smon zuständig.

Für diejenigen Leser, die nicht das 2. Schneider-Sonderheft besitzen, in dem bereits die Anleitung zum Smon abgedruckt wurde, haben wir in Bild 5 alle Befehle in Kurzform zusammengefaßt. Die Befehle werden in der Regel durch Eingabe des Kennbuchstabens und einer hexadezimalen Adresse, die vier Ziffern lang ist, aktiviert.

Folgende Befehle, die insbesondere für die EPROM-Programmierung sehr wichtig sind, bedürfen einer ausführlicheren Erklärung:

Der Befehl

```
X ⟨start⟩ ⟨ende⟩ ⟨ziel⟩
```

verschiebt den Speicherbereich von der Adresse <start> bis <ende> nach <ziel>.

Der Befehl

```
Z ⟨start⟩ ⟨ende⟩ ⟨byte⟩
```

füllt den Speicherbereich von der Adresse <start> bis <ende> mit dem Wert <byte>.

Der Befehl

```
: ⟨byte⟩
```

bestimmt die Speicherkonfiguration des CPC. Wenn Sie für <byte> die Werte 00 bis FB (hex) einsetzen, wird das Erweiterungs-ROM mit der entsprechenden Nummer in den Speicherbereich von C000 bis FFFF (hex) eingeblendet. Der Befehl »: 07« schaltet beispielsweise das Disketten-ROM ein.

Die Werte von FC bis FF (hex) wählen die interne Speicherverteilung des CPC nach folgendem Muster aus:

```
FC: Unteres und oberes ROM
    eingeschaltet.
FD: Unteres ROM aus- und oberes
    ROM eingeschaltet.
FE: Unteres ROM ein- und oberes
    ROM ausgeschaltet.
FF: Unteres und oberes ROM
    ausgeschaltet.
```

Die Speicherverteilung nach dem Muster FF (hex) ist die Standardkonfiguration des CPC.

Auf Wunsch schicken wir Ihnen gegen einen mit 1,30 Mark frankierten und an Sie selbst adressierten Rückumschlag auch gerne die ausführliche Anleitung von SMON zu.

Die oben besprochenen Befehle von SMON zeigen zugleich, wie man beispielsweise das Betriebssystem des CPC ausliest, um es in einer geänderten Form in ein EPROM zu programmieren.

Mit

```
: FE
X 0000 3FFF 4000
```

kopieren Sie den Inhalt des unteren ROM-Bereichs ab der Adresse 4000 (hex) in den Arbeitsspeicher. Nun können Sie die gewünschten Änderungen vornehmen und darauf mit

```
J 9000
4
1
4000 7FFF 0000
```

die gepatchte Version in die untere Hälfte eines 27256A (kein A-Typ: Menüpunkt 5 anwählen!) programmieren.

Bauteil	Bezeichnung	Wert/Typ
	3 IC-Sockel	14polig
	1 IC-Sockel	20polig
	1 IC-Sockel	40polig
	Textool-Sockel	28polig
	Lochstreifenrasterplatine oder IC-Experimentierpaltine	
R1	Potentiometer	2 KΩ
R2	Widerstand	1 KΩ
R3	Widerstand	33 KΩ
R4	Widerstand	6,8 Ω
R5	Widerstand	1 Ω
R6	Widerstand	22 KΩ
R7	Widerstand	10 KΩ
R8	Widerstand	22 KΩ
R9	Widerstand	10 KΩ
R10	Widerstand	47 KΩ
C1	Elektrolytkondensator	220 μF
C2	Keramikkondensator	100 nF
C3	Elektrolytkondensator	220 μF
C4	Kondensator	100 pF
C5	Elektrolytkondensator	68 μF
C6	Keramikkondensator	100 nF
L	Schalenkernspule	250 AI
D1	Zenerdiode	ZPD 22 Volt
D2	Zenerdiode	ZPD 9,1 Volt
D3	Germaniumdiode	BB 119
D4	Siliziumdiode	1N 4148
T1	PNP-Transistor	BC 307 oder BC 558
T2	PNP-Transistor	BC 307 oder BC 558
IC1	Parallelportbaustein	8255 PIO
IC2	Spannungswandler	TL 497
IC3	Sechs Inverter mit offenem Kollektor	74LS06
IC4	8-Bit-Register	74HC573
IC5	Vier ODER-Gatter	74LS32

Tabelle. Diese Bauteile benötigen Sie zum Bau des Happy-EPROMers

Entsprechend lesen Sie mit

```
:FD
X C000 FFFF 4000
```

den Inhalt des oberen ROM-Bereichs in den Arbeitsspeicher und programmieren ihn nach einer eventuellen Änderung durch

```
J 9000
4
1
4000 7FFF 4000
```

in das EPROM.

Sie sollten sich gleich angewöhnen, nur den CPC-Speicherbereich von 4000 bis 7FFF (hex) für die Daten des EPROM zu verwenden, weil ab Adresse 8000 (hex) der Maschinencode von Smon folgt und sich vor der Adresse 4000 (hex) residente Basic-Programme befinden könnten. Wenn Sie keine Basic-Programme benötigen, dürfen Sie als niedrigste Adresse den Wert 170 (hex) verwenden.

Selbstverständlich können Sie auch kleinere Datenmengen als 16 KByte programmieren, indem Sie vorher einen kleineren Arbeitsspeicherbereich (zum Beispiel 5000 bis 5400 (hex)) angeben.

Wie bereits ausgeführt, erlaubt der Happy-EPROMer auch das Patchen des Betriebssystems Ihres CPC. Die Möglichkeiten, die sich auf diesem Gebiet ergeben, sind vielfältig und nahezu unbegrenzt.

Wie wäre es beispielsweise mit einer individuellen Einschaltmeldung, mit deutschen Fehlermeldungen oder einer Reset-Routine, die den Arbeitsspeicher nicht löscht?

Wir sind gespannt, welche Ideen Sie entwickeln werden. Wenn Sie der Meinung sind, daß Sie einen interessanten Patch für das Betriebssystem geschrieben haben, dann schicken Sie ihn uns doch mit einer Beschreibung zu. Wir werden dann die besten Patches für unsere Leser veröffentlichen (selbstverständlich gegen ein angemessenes Honorar).

Um Ihnen deprimierende Mißerfolge beim Patchen des Betriebssystems zu ersparen, darf ein Problem nicht unerwähnt bleiben: Für die Neuprogrammierung des CPC-Betriebssystems lassen sich nicht alle EPROMs verwenden, weil die Bausteine einiger Hersteller zwar außerhalb des Computers als Erweiterungs-ROMs brav ihren Dienst tun, im Computer jedoch wegen zu geringer Ausgangsleistungen versagen. Dieser Umstand zeigte sich bei Experimenten an vielen verschiedenen CPCs.

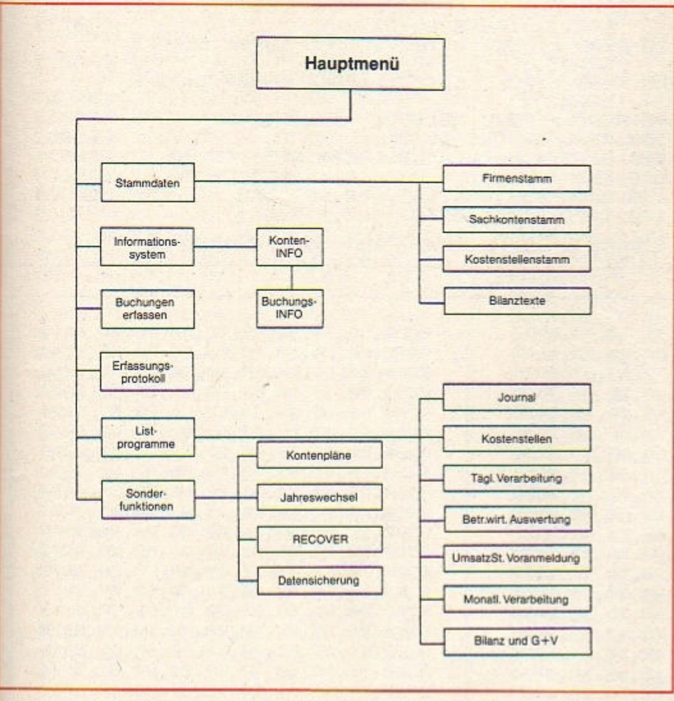
Die EPROMs von Mitsubishi und den 27256-Typ von SGS können Sie dagegen für einen Betriebssystem-Patch bedenkenlos einsetzen.

(Peter Bündgens/ma)

Praxiserprobte Finanzbuchhaltung und Fakturierung

für den Einsatz im Klein- und Mittelbetrieb

»Finanzbuchhaltung«, eine praxiserprobte Sachkontenbuchhaltung mit Kostenstellenrechnung, ist ein menügesteuertes und bedienerfreundliches Programmpaket. Sie können mit ihm schnell und unkompliziert ein EDV-unterstütztes Informationssystem in Ihrem Betrieb installieren.
Per Tastendruck können Sie sich jederzeit über die Finanzlage informieren. Das mitgelieferte ausführliche Handbuch erklärt Ihnen anhand zahlreicher Buchungsbeispiele und Abbildungen die einfache Bedienung des Programms.

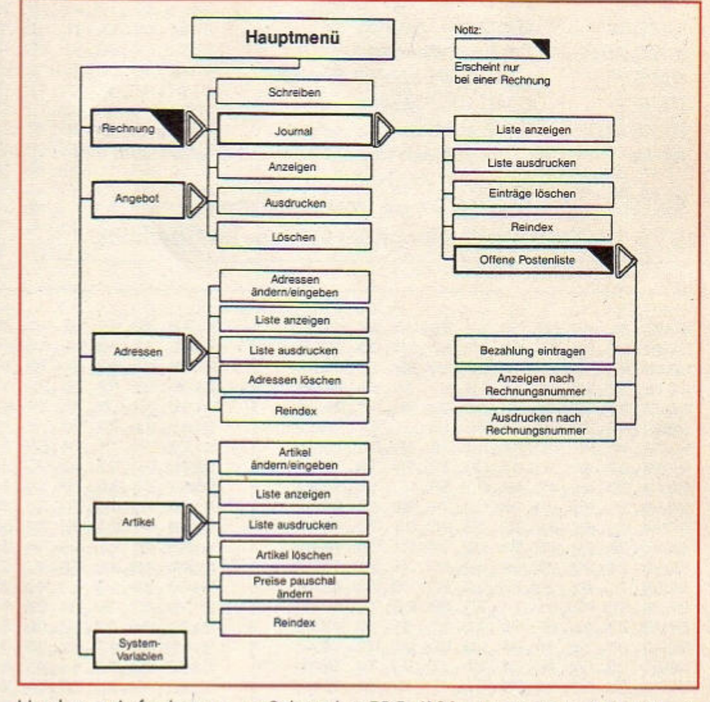


Hardware-Anforderung: • Schneider Joyce PCW 8256 mit einem Laufwerk oder • Schneider Joyce PCW 8512 und PCW 8256 mit zwei Laufwerken oder • Schneider CPC 6128 und externes Laufwerk

Best.-Nr. 51618/51623/51615 **je DM 194,–***

*inkl. MwSt. Unverbindliche Preisempfehlung

»Fakturierung«: Das dBASE-II-Anwenderprogramm für den Klein- und Mittelbetrieb unterstützt und vereinfacht Ihre Routinetätigkeiten: • Angebotsschreibung • Rechnungsschreibung (mit automatischem Abbuchen aus der Artikeldatei) • Offene-Posten-Verwaltung • Adreßverwaltung • Artikelverwaltung. Programmspezifische Eigenschaften: • Kooperation einzelner Komponenten miteinander • Individuelle Anpassung an Ihre Bedürfnisse möglich • Ein sehr ausführliches Handbuch mit vielen Grafiken ist im Lieferumfang enthalten.



Hardware-Anforderung: • Schneider CPC 6128 mit externem 3"-Laufwerk • Schneider Joyce PCW 8256 mit einem Laufwerk oder • Schneider Joyce PCW 8256 mit zwei Laufwerken und Schneider Joyce PCW 8512
Software-Anforderung: Für den Einsatz der Fakturierung ist das dBASE-II-Datenbanksystem, Verlag Markt&Technik, erforderlich.

Best.-Nr. 51619/51624/51616 **je DM 94,–***

*inkl. MwSt. Unverbindliche Preisempfehlung

	Version	Format	Bestell-Nr.	DM	sFr	öS.
Finanz-buch-haltung	Joyce PCW 8256 mit einem Laufwerk	3"	51618	194,–*	175,–	1940,–*
	Joyce PCW 8512 und PCW 8256 mit zwei Laufwerken	3"	51623	194,–	175,–	1940,–*
	CPC 6128 und externes Laufwerk	3"	51615	194,–*	175,–	1940,–*
Faktu-rierung	Joyce PCW 8256 mit einem Laufwerk	3"	51619	94,–*	82,–	940,–*
	Joyce PCW 8512 und Joyce PCW 8256 mit zwei Laufwerken	3"	51624	94,–*	82,–	940,–*
	CPC 6128 mit externem Laufwerk	3"	51616	94,–*	82,–	940,–*
dBASE II	CPC 6128	3"	50304	199,–*	178,–	1990,–*
	Joyce	3"	50305	199,–*	178,–	1990,–*

*inkl. MwSt. Unverbindliche Preisempfehlung

Dieses Markt&Technik-Softwareprodukt erhalten Sie in den Fachabteilungen der Warenhäuser, im Versandhandel, in Computer-Fachgeschäften oder im Buchhandel. Wenn Sie direkt beim Verlag bestellen wollen: gegen Vorauskasse durch Verrechnungsscheck oder mit der abgedruckten Zahlkarte.

Markt&Technik Verlag AG, Buchverlag, Hans-Pinsel-Straße 2, 8013 Haar bei München, Telefon (089) 4613-0

Bestellungen im Ausland bitte an: SCHWEIZ: Markt&Technik Vertriebs AG, Kollerstrasse 3, CH-6300 Zug, Telefon (042) 415656 · ÖSTERREICH: Rudolf Lechner & Sohn, Heizwerkstraße 10, A-1232 Wien, Telefon (0222) 677526 · Ueberreuter Media Verlagsges. mbH (Großhandel), Alser Straße 24, A-1091 Wien, Telefon (0222) 481538-0.

703234

Befehlsübersicht Supermon CPC-1002

A	ASCII-Dump	!	Bye (Monitor verlassen)
B	Breakpoint setzen	:	RAM/ROM-Bank wählen
C	Copy (Datei kopieren)	I	Externes Kdo. aufrufen
D	Disassembler	@	CATalog-Funktion
F	Find (Bytefolgen suchen)	#	Drucker ein
G	Go (Routine ausführen)	&	40/80 Zeichen umschalten
H	Header anzeigen	(	User Jump definieren
J	Jump (Programm aufrufen)	)	User Jump anzeigen
K	Kill (Breakpoint löschen)	0	SPEED 0 wählen (Kassette)
L	Load (Programm laden)	1	SPEED 1 wählen (Kassette)
M	Memory Dump	.	7/8-Bit-Modus umschalten
O	On/Off (Kass.Motor ein/aus)	+	Hexadezimal addieren
R	Register anzeigen	–	Hexadezimal subtrahieren
S	Save (Programm speichern)	↑	Sprungdistanz berechnen
T	Text eingeben	X	Speicherbereich verschieben
U	User Jump ausführen	Y	Yank (Progr. verschob. laden)
W	Write (Bytes schreiben)	Z	Fill (Speicherber. belegen)

Esc	Funktion abbrechen	IBASIC	Basic-Kaltstart
Enter	Monitor-Warmstart	Ctrl C	Monitor-Kaltstart

Externe DOS-Kommandos (nur bei angeschlossenem Laufwerk)

IA	Aktuelles Laufwerk ist Laufwerk A
IB	Aktuelles Laufwerk ist Laufwerk B
IDRIVE	Aktuelles Laufwerk wählen
IDIR	Directory auflisten
IERA	Datei löschen
IREN	Datei umbenennen
IDISC	LOAD/SAVE von/auf Diskette
IDISC.IN	LOAD von Diskette
IDISC.OUT	SAVE auf Diskette
ITAPE	LOAD/SAVE von/auf Kassette
ITAPE.IN	LOAD von Kassette
ITAPE.OUT	SAVE auf Kassette
ICPM	CP/M aufrufen (danach kann mit S-MON nicht mehr gearbeitet werden)
IUSER	(Von S-MON aus nicht verwendbar)

Bild 5. Die Befehle des Monitors »Smon« im Überblick

```
10 '***************************************
   ***                                    [431C]
20 '*         EPROMMER - PROGRAMM
    *                                     [96EC]
30 '*
    *                                     [663C]
40 '*  (C) 05/1987 by   Peter Buendgens
    *                                     [96AC]
50 '*                   Eschstrasse  31
    *                                     [3CD8]
60 '*                   5160 Dueren
    *                                     [AF20]
70 '* Tel. 02421-56567, werkt. ab 18-19
   h *                                    [D234]
80 '*
    *                                     [1446]
90 '***************************************
   ***                                    [F32C]
100 MODE 2                                [9C50]
110 IF HIMEM>36863 THEN MEMORY &8FFF:LOA
    D"eprom.bin",&9000                    [39E8]
120 PRINT". SMON aktivieren (j/n) ?"      [F370]
130 a$=INKEY$:IF a$="j"THEN GOSUB 150 EL
    SE IF a$<>"n"THEN 130                 [6C48]
140 CALL &9000:GOTO 110                   [010A]
150 PRINT"Rueckkehr mit dem Befehl >>J 9
    000<< !"                              [72B2]
160 IF HIMEM>32767 THEN MEMORY &7FFF:LOA
    D"smon.bin",&8000                     [8920]
170 CALL &8000                            [55C4]
180 RETURN                                [3A32]
```

Listing 3. Mit »Promon« können Sie starten

```
10 MEMORY &7FFF:IF PEEK(&8000)<>49 THEN
   LOAD"smon.bin",&8000                   [19DC]
20 IF PEEK(6)=128 THEN RESTORE 130        [FDF8]
30 IF PEEK(6)=&7B THEN RESTORE 140        [A304]
40 IF PEEK(6)=&91 THEN 120                [615C]
50 READ z:POKE &81DE,z                    [7E04]
60 READ z:POKE &82C8,z:POKE &8573,z:POKE
    &8817,z:POKE &89B2,z                  [C012]
70 READ z:POKE &81E2,z:POKE &89B8,z:POKE
    &8A01,z                               [FAC2]
80 READ z:POKE &81E3,z:POKE &89B9,z:POKE
    &8A02,z                               [0ACA]
90 READ z:POKE &8100,z                    [39BA]
100 READ z:POKE &8101,z                   [610C]
110 SAVE"smon.bin",B,&8000,&F5A,&8000     [7BF2]
120 END                                   [AB14]
130 DATA &5E,&A4,&86,&B2,&07,&B8          [00C0]
140 DATA &5B,&A8,&27,&B7,&1F,&B1          [01D6]
```

Listing 4. Die Routine »Patch« paßt »Smon« an

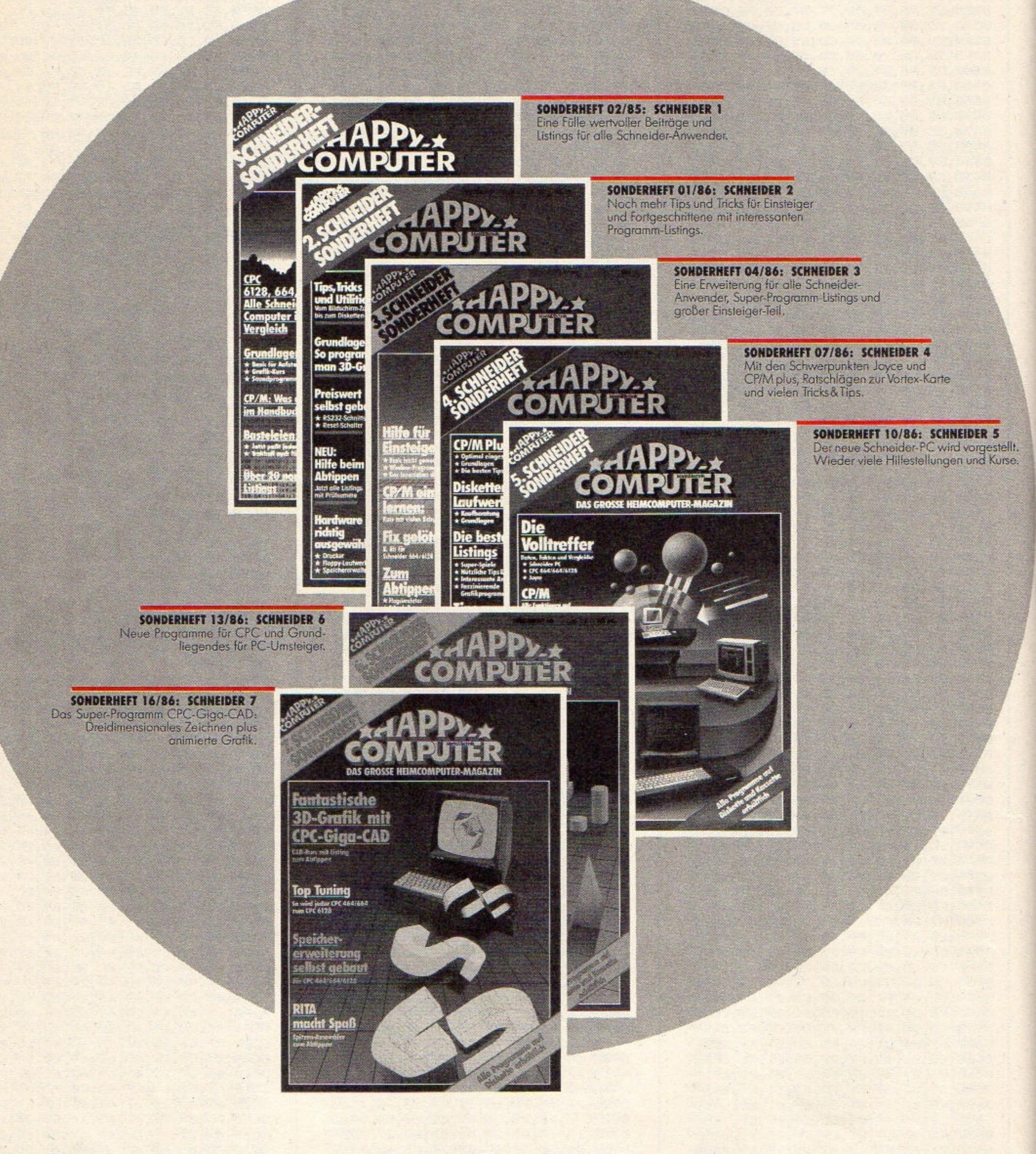
```
8000,31,00,C0,CD,00,BB,CD,4E,0F68
8008,BB,CD,FF,BB,CD,65,BC,CD,7CD9
8010,A7,BC,0E,00,CD,15,B9,7C,7A72
8018,32,2B,81,3A,1D,81,3D,E6,0230
8020,01,3C,32,FD,80,CD,0E,BC,0184
8028,AF,32,FC,80,3A,1E,81,47,4C6D
8030,4F,AF,CD,32,BC,3A,1F,81,1377
8038,47,4F,3E,01,CD,32,BC,3A,3072
8040,20,81,47,4F,CD,38,BC,11,3BB1
8048,2E,81,CD,2D,82,31,00,C0,2824
8050,CD,E0,81,AF,32,FC,80,3E,470E
8058,3E,CD,FE,81,CD,44,82,CD,3D21
8060,74,82,21,8A,80,4F,23,23,1359
8068,7E,B7,28,F3,23,B9,20,F6,1BBA
8070,CD,FE,81,11,7C,80,D5,7E,4804
8078,23,66,6F,E9,11,8F,81,30,08F6
8080,03,11,94,81,CD,E0,81,CD,1B77
8088,2D,82,18,C1,21,20,83,3A,39A4
8090,5C,83,41,F0,83,4D,0E,84,0C54
8098,57,39,84,48,5A,84,4C,9A,3102
80A0,84,7C,80,83,40,36,85,53,46B1
80A8,F0,84,43,3E,85,59,DC,84,56F0
80B0,23,49,83,47,9D,85,26,39,1159
80B8,83,54,D4,85,44,FD,87,58,46D2
80C0,B3,85,42,F8,85,52,65,86,7A2C
80C8,4B,A9,86,46,BB,86,2B,70,1C86
80D0,87,2D,89,87,5E,93,87,55,5077
80D8,B5,87,4A,B0,87,28,C0,87,7C9F
80E0,29,C9,87,30,D4,87,31,DB,3125
80E8,87,5A,E5,87,0D,4D,80,03,418F
80F0,00,80,4F,74,BC,2E,2F,83,2B25
80F8,3F,80,88,00,00,00,00,00,2E80
8100,1F,B1,00,00,00,00,00,00,23C0
8108,00,00,00,00,00,00,00,00,0000
8110,00,00,00,00,00,00,00,00,0000
8118,00,00,00,00,00,02,00,1A,0012
8120,00,00,00,00,00,00,00,00,0000
8128,00,00,00,00,08,AD,20,20,0294
8130,20,20,20,20,20,2A,2A,20,1FDC
8138,53,55,50,45,52,4D,4F,4E,31E4
8140,20,43,50,43,2D,31,30,30,0F0C
8148,32,20,2A,2A,0A,0A,0D,20,16A2
8150,20,43,6F,70,79,72,69,67,0895
8158,68,74,20,28,63,29,20,31,2C4D
8160,39,38,35,20,62,79,20,48,14DC
8168,61,70,70,79,20,43,6F,6D,25AF
8170,70,75,74,65,72,0A,0A,0D,2E31
8178,00,25,25,25,20,56,6F,6C,0F5A
8180,6B,65,72,20,45,76,65,72,23C8
8188,74,73,20,25,25,25,00,4E,2162
8190,4F,54,20,07,4F,4B,00,4C,35E8
8198,6F,61,64,3A,00,53,5A,2D,2135
81A0,48,2D,50,4E,43,20,20,41,2339
81A8,20,20,42,43,20,20,20,44,15F4
81B0,45,20,20,20,48,4C,20,20,2F90
81B8,20,49,58,20,20,20,49,59,0A0B
81C0,00,42,52,45,41,4B,20,07,1DF3
81C8,00,20,46,69,6C,65,6E,61,0419
81D0,6D,65,20,3F,20,00,3A,2B,296F
81D8,81,B7,CA,3A,BD,C3,5E,BD,7145
81E0,F5,3A,27,B7,B7,20,03,F1,7E5F
81E8,C9,F5,3E,0A,CD,FE,81,3E,5A0C
81F0,0D,CD,FE,81,F1,C9,E6,FF,278F
81F8,FE,20,30,02,3E,2E,F5,3A,71B8
8200,FC,80,B7,28,23,F1,F5,FE,49A8
8208,0D,28,0C,FE,0A,28,08,CB,02CB
8210,BF,FE,20,30,02,3E,2E,C5,6771
8218,06,20,4F,79,CD,2B,BD,38,02F6
8220,05,10,F8,AF,32,FC,80,C1,10D1
8228,F1,CD,5A,BB,C9,F5,D5,1A,4F1C
8230,B7,28,06,CD,FE,81,13,18,585A
8238,F6,D1,F1,C9,F5,3E,20,CD,5A2D
8240,FE,81,F1,C9,F5,3E,07,CD,4A63
8248,5A,BB,F1,C9,7C,CD,51,82,1184
8250,7D,F5,0F,0F,0F,0F,CD,5A,0354
8258,82,F1,E6,0F,FE,0A,38,02,66DA
8260,C6,07,C6,30,CD,FE,81,C9,7D5B
8268,CD,8A,BB,CD,06,BB,F5,CD,5C4B
8270,8D,BB,F1,C9,CD,68,82,FE,7CC2
8278,61,D8,FE,7B,D0,D6,20,C9,1BA1
8280,CD,74,82,FE,FC,C8,FE,0D,6111
8288,C8,D6,30,38,F3,FE,0A,38,504C
8290,0B,D6,07,FE,10,30,E9,FE,3E6C
8298,0A,3F,30,E4,F5,CD,5A,82,062A
82A0,F1,C9,CD,3C,82,CD,80,82,5606
82A8,D0,C5,07,07,07,07,4F,CD,59A7
82B0,80,82,30,02,B1,37,C1,C9,62BF
82B8,CD,BD,82,D0,EB,CD,A2,82,512A
82C0,D0,67,CD,A5,82,6F,C9,21,662F
82C8,A8,AC,3A,2B,81,B7,28,03,7C77
82D0,21,8A,AC,36,00,CD,D6,81,26F9
82D8,D0,E5,AF,06,FF,04,BE,23,4277
82E0,20,FB,E1,37,C9,3A,FC,80,3648
82E8,06,10,B7,C0,3A,FD,80,0F,1ECB
82F0,D0,06,08,C9,CD,09,BB,3F,6315
82F8,D8,FE,0D,28,08,FE,20,28,5370
8300,04,FE,FC,20,EF,CD,68,82,241E
8308,FE,FC,C8,37,C9,CD,E0,81,5E4D
8310,CD,4C,82,CD,3C,82,CD,3C,6B5E
8318,82,C9,7C,BA,C0,7D,BB,C9,712B
8320,3E,FD,CD,60,83,AF,CD,60,38BE
8328,83,CD,E0,81,C3,64,C0,3A,60E2
8330,F7,81,EE,80,32,F7,81,37,4D79
8338,C9,3A,FD,80,3D,20,02,3E,7CF2
8340,02,32,FD,80,CD,0E,BC,37,1D3F
8348,C9,3E,01,32,FC,80,CD,2E,6C54
8350,BD,3F,D0,3E,0D,CD,2B,BD,4B17
8358,E1,C3,5F,80,CD,A2,82,D0,4694
8360,FE,FC,30,06,4F,CD,0F,B9,478B
8368,37,C9,0F,F5,DC,09,B9,F1,2037
8370,F5,D4,06,B9,F1,0F,F5,DC,4252
8378,03,B9,F1,D4,00,B9,37,C9,3EE3
8380,CD,3C,82,3E,FF,32,E8,B4,7C74
8388,CD,C7,82,D0,78,B7,C8,E5,4A69
8390,48,06,00,3E,2C,ED,B1,E2,2534
```

Listing 1. Die hexadezimalen Werte zu »Smon« (Eingabe mit dem Programm »CPC«)

```
8398,9B,83,2B,2B,CB,FE,1E,00,6F0C
83A0,DD,21,0A,81,23,BE,20,15,6C25
83A8,23,FD,21,02,81,CD,CA,83,2CEB
83B0,30,0B,DD,2B,DD,2B,FD,21,044F
83B8,05,81,CD,CA,83,E1,D5,CD,313B
83C0,D4,BC,D1,D0,7B,CD,1B,00,52FA
83C8,37,C9,FD,36,00,FF,FD,75,3773
83D0,01,FD,74,02,1C,FD,E5,E3,335D
83D8,DD,75,00,DD,74,01,E1,2B,7C5D
83E0,23,FD,34,00,7E,B7,C8,FE,2802
83E8,2C,20,F5,36,00,23,37,C9,03EB
83F0,CD,BD,82,D0,CD,E5,82,CB,50B3
83F8,20,CB,20,C5,CD,0D,83,7E,2DB4
8400,CD,F6,81,23,10,F9,C1,CD,4B3B
8408,F4,82,38,EF,3F,C9,CD,BD,508B
8410,82,D0,CD,E5,82,C5,CD,0D,6463
8418,83,E5,C5,7E,CD,51,82,CD,6165
8420,3C,82,23,10,F6,C1,E1,CD,3E5B
8428,3C,82,7E,CD,F6,81,23,10,3872
8430,F9,C1,CD,F4,82,38,DE,3F,5F53
8438,C9,CD,BD,82,D0,CD,E0,81,4CB5
8440,CD,E5,82,C5,CD,0D,83,3E,44B4
8448,08,CD,FE,81,CD,A2,82,30,2544
8450,04,77,23,10,F7,C1,38,EB,1E87
8458,3F,C9,2A,00,81,CD,E0,81,2EFD
8460,E5,DD,E1,CD,67,85,CD,2D,55AB
8468,82,DD,7E,12,CD,E0,81,CD,7C87
8470,51,82,CD,3C,82,DD,6E,15,15CD
8478,DD,66,16,CD,4C,82,DD,5E,789C
8480,18,DD,56,19,19,2B,CD,3C,31D2
8488,82,CD,4C,82,DD,6E,1A,DD,7459
8490,66,1B,CD,3C,82,CD,4C,82,289E
8498,37,C9,CD,A0,84,D0,18,23,3D13
84A0,11,C9,81,CD,2D,82,CD,C7,240D
84A8,82,D0,AF,CD,6B,BC,11,00,6DBA
84B0,08,CD,77,BC,30,21,22,00,3320
84B8,81,CD,5A,84,DD,6E,15,DD,7767
84C0,66,16,C9,CD,E0,81,E5,CD,2773
84C8,83,BC,E1,30,0A,DD,75,15,727B
84D0,DD,74,16,CD,7A,BC,D8,CD,7DCD
84D8,7D,BC,B7,C9,CD,A0,84,D0,0EC0
84E0,CD,E0,81,11,97,81,CD,2D,48BB
84E8,82,CD,BD,82,30,E9,18,D3,6F07
84F0,11,C9,81,CD,2D,82,CD,C7,240D
84F8,82,D0,11,00,08,CD,8C,BC,75F0
8500,30,2F,AF,CD,6B,BC,CD,E0,0A22
8508,81,CD,A5,82,30,23,ED,47,6FD1
8510,CD,B8,82,30,1C,B7,ED,52,5874
8518,3F,30,16,23,EB,E5,CD,BD,169B
8520,82,44,4D,E1,30,0B,ED,57,5791
8528,CD,98,BC,30,04,CD,8F,BC,56B6
8530,D8,CD,92,BC,B7,C9,11,00,407E
8538,08,CD,9B,BC,37,C9,CD,67,2C81
8540,85,EB,11,00,08,CD,8C,BC,78B0
8548,30,E7,DD,2A,00,81,DD,66,3B18
8550,16,DD,6E,15,DD,56,19,DD,378F
8558,5E,18,DD,46,1B,DD,4E,1A,35EA
8560,DD,7E,12,CB,87,18,C1,C5,7AEF
8568,E5,2A,00,81,7E,FE,20,30,7068
8570,01,23,11,A8,AC,D5,06,08,06D0
8578,CD,8F,85,30,09,3E,2E,12,561E
8580,13,06,03,CD,8F,85,AF,12,0390
8588,7B,D1,93,E1,C1,47,C9,7E,1348
8590,B7,C8,FE,20,28,02,12,13,753F
8598,23,10,F4,37,C9,CD,BD,82,0CF4
85A0,D0,CD,E0,81,11,AD,85,D5,4CB3
85A8,E5,CD,1C,86,C9,F5,CD,35,4E13
85B0,86,F1,C9,CD,B8,82,D0,B7,6C6F
85B8,ED,52,3F,D0,44,4D,CD,BD,6AD3
85C0,82,D0,CD,1A,83,38,08,09,69E1
85C8,EB,09,03,ED,B8,37,C9,EB,7D15
85D0,03,ED,B0,C9,CD,BD,82,D0,2518
85D8,CD,E5,82,CB,20,CB,20,48,4114
85E0,CD,0D,83,CD,68,82,FE,FC,7938
85E8,3F,C8,FE,0D,3F,C8,77,CD,306B
85F0,F6,81,23,10,EE,41,18,E8,588C
85F8,CD,BD,82,D0,7C,B5,C8,ED,54C9
8600,5B,0D,81,7A,B3,C0,E7,32,3E24
8608,0C,81,36,F7,22,0D,81,3E,2FE8
8610,C3,21,4E,86,32,30,00,22,6912
8618,31,00,37,C9,ED,4B,11,81,1417
8620,ED,5B,13,81,2A,0F,81,E5,6ABB
8628,F1,2A,15,81,DD,2A,17,81,7E5F
8630,FD,2A,19,81,C9,ED,43,11,7A5B
8638,81,ED,53,13,81,22,15,81,74BB
8640,F5,E1,22,0F,81,DD,22,17,415F
8648,81,FD,22,19,81,C9,CD,35,7C93
8650,86,E1,2B,CD,E9,81,11,C1,775F
8658,81,CD,2D,82,CD,4C,82,CD,78D1
8660,65,86,C3,7C,80,CD,E0,81,0AD5
8668,CD,E9,81,11,9D,81,CD,2D,4AAB
8670,82,CD,E0,81,3A,0F,81,06,66B8
8678,08,4F,3E,30,CB,21,CE,00,1440
8680,CD,FE,81,10,F5,CD,3C,82,4C46
8688,3A,10,81,CD,51,82,21,11,0523
8690,81,06,05,5E,23,56,23,CD,448B
8698,3C,82,EB,CD,4C,82,EB,10,2E9E
86A0,F2,CD,E0,81,CD,E9,81,37,5AA9
86A8,C9,2A,0D,81,7C,B5,C8,3A,672E
86B0,0C,81,77,21,00,00,22,0D,2AF9
86B8,81,37,C9,CD,B8,82,D0,B7,5E6F
86C0,ED,52,3F,D0,23,44,4D,21,6853
86C8,21,81,C5,CD,37,87,C1,D0,2646
86D0,DD,21,21,81,EB,CD,E0,81,6FDD
86D8,DD,7E,00,B7,C8,DD,7E,01,7FB9
86E0,ED,B1,37,E0,E5,FD,E1,CD,57F3
86E8,F4,82,3F,D8,DD,7E,02,B7,5743
86F0,28,08,DD,7E,03,FD,BE,00,08D0
86F8,20,E3,DD,7E,04,B7,28,08,3624
8700,DD,7E,05,FD,BE,01,20,D5,7B11
8708,DD,7E,06,B7,28,08,DD,7E,7A14
8710,07,FD,BE,02,20,C7,2B,CD,29A7
8718,4C,82,CD,3C,82,7E,CD,51,18C3
8720,82,23,7E,CD,51,82,FD,7E,4BD4
8728,01,CD,51,82,FD,7E,02,CD,3719
8730,51,82,CD,E0,81,18,A6,06,1A82
8738,04,36,FF,23,CD,3C,82,CD,1501
8740,80,82,30,1B,07,07,07,07,671D
8748,E6,F0,77,CD,80,82,30,07,4B5F
8750,B6,77,23,10,E4,37,C9,FE,4530
8758,FC,C8,3E,08,CD,FE,81,FE,4F2C
8760,FC,C8,3E,58,CD,FE,81,CD,4A1F
8768,FE,81,2B,36,00,23,18,E2,591E
8770,CD,B8,82,D0,19,CD,7D,87,5641
8778,CD,4C,82,37,C9,CD,3C,82,6336
8780,3E,3D,CD,FE,81,CD,3C,82,01C6
8788,C9,CD,B8,82,EB,D0,B7,ED,4D7B
8790,52,18,E2,CD,B8,82,D0,B7,394F
8798,ED,52,2B,2B,7C,FE,FF,28,641E
87A0,02,B7,C0,AD,E6,80,C0,CD,3A6D
87A8,7D,87,7D,CD,51,82,37,C9,1C17
87B0,CD,BD,82,D0,E9,CD,E0,81,51BD
87B8,2A,29,81,7C,B5,C8,37,E9,0EAF
87C0,CD,BD,82,D0,22,29,81,37,5401
87C8,C9,CD,3C,82,2A,29,81,CD,585B
87D0,4C,82,37,C9,3E,19,21,4D,0D6B
87D8,01,18,05,3E,32,21,A7,00,059A
87E0,CD,68,BC,37,C9,CD,B8,82,6CFE
87E8,D0,B7,ED,52,3F,D0,CD,A2,5EC0
87F0,82,EB,13,4F,71,23,1B,7A,7E18
87F8,B3,20,F9,37,C9,CD,BD,82,4954
8800,D0,CD,F4,82,3F,D8,CD,0D,4EEF
8808,83,11,08,AD,ED,53,2C,81,48ED
8810,0E,00,CD,03,89,FD,21,A8,19C6
8818,AC,FE,DD,20,04,CB,C1,18,72B6
8820,F1,FE,FD,20,04,CB,C9,18,5826
8828,E9,FE,ED,28,68,FE,CB,CA,55C4
8830,BA,88,CB,7F,20,30,CB,77,61B1
8838,20,08,11,E6,8A,CD,1B,89,199B
8840,18,BF,11,A2,8A,47,CD,62,2F74
8848,8A,FD,36,00,20,FD,23,EB,7ED9
8850,78,0F,0F,0F,CD,78,8A,FD,38B1
8858,36,00,2C,FD,23,78,CD,78,104A
8860,8A,CD,AC,89,18,9B,CB,77,681D
8868,20,20,11,AD,8A,47,0F,0F,15AD
8870,0F,E6,07,CD,8F,8A,EB,78,35CE
8878,CD,78,8A,CD,AC,89,18,81,62E5
8880,11,79,81,CD,E0,81,CD,2D,0E83
8888,82,C9,11,5C,8C,E6,3F,CD,73EB
8890,1B,89,C3,01,88,CD,03,89,304B
8898,D6,40,38,18,11,C3,8D,FE,7FE0
88A0,3C,38,0B,D6,60,38,0D,11,1FEB
88A8,0E,8F,FE,1C,30,06,CD,1B,3AD9
88B0,89,C3,01,88,CD,57,8A,C3,7A03
88B8,01,88,CD,03,89,47,E6,C0,3F48
88C0,20,17,78,11,C5,8A,0F,0F,1FC1
88C8,0F,E6,07,CD,8F,8A,78,CD,345D
88D0,78,8A,EB,CD,AC,89,C3,01,09F3
88D8,88,11,DD,8A,17,17,17,E6,536C
88E0,03,3D,CD,8F,8A,78,0F,0F,1A31
88E8,0F,E6,07,C6,30,FD,77,00,301A
88F0,FD,23,FD,36,00,2C,FD,23,6BE9
88F8,78,CD,78,8A,EB,CD,AC,89,0D5D
8900,C3,01,88,7E,23,F5,E5,2A,740C
8908,2C,81,FE,20,36,2E,38,01,2AF9
8910,77,23,22,2C,81,E1,CD,51,3387
8918,82,F1,C9,EB,47,B7,28,08,6A6C
8920,23,7E,FE,60,38,FA,10,F8,1530
8928,7E,FE,7B,20,05,CD,57,8A,0ED8
8930,18,7A,D6,20,06,05,FD,77,0BE9
8938,00,FD,23,23,05,7E,FE,60,395C
8940,30,6A,FE,20,20,F0,FD,77,1C0D
8948,00,FD,23,10,F9,FD,2B,FD,3EB7
8950,77,00,FD,23,23,7E,FE,60,276C
8958,30,52,FE,5E,28,44,FE,21,172D
8960,28,45,FE,2A,28,12,FE,2B,18FF
8968,28,1A,FE,40,20,E1,EB,FD,0AEF
8970,77,00,FD,23,CD,03,89,EB,218D
8978,FD,77,00,FD,23,EB,CD,03,6FBD
8980,89,EB,18,CB,FD,36,00,40,7680
8988,FD,23,EB,CD,03,89,EB,E5,64FF
8990,6F,26,00,07,30,02,26,FF,3F4B
8998,19,FD,75,00,FD,23,7C,E1,3A1D
89A0,18,AD,CD,11,8A,18,A8,CD,3A5D
89A8,2C,8A,18,A3,FD,36,00,00,3A80
89B0,FD,21,A8,AC,CD,3C,82,3A,6EA6
89B8,27,B7,FE,10,38,F6,FD,7E,231C
89C0,00,FD,23,B7,28,2C,FE,40,301C
89C8,28,0F,FE,2A,28,1A,FE,20,0A54
89D0,30,02,3E,25,CD,FE,81,18,199A
89D8,E5,FD,6E,00,FD,66,01,CD,46BF
89E0,4C,82,FD,23,FD,23,18,D6,1C92
89E8,FD,7E,00,CD,51,82,FD,23,6C89
89F0,18,CC,E5,2A,2C,81,01,08,226E
89F8,AD,B7,ED,42,EB,CD,3C,82,6656
8A00,3A,27,B7,FE,23,38,F6,0A,0DDE
8A08,CD,F6,81,03,1D,20,F8,D1,4A59
8A10,C9,E5,26,58,2E,49,CB,41,5D03
8A18,20,0A,26,59,CB,49,20,04,14E8
8A20,26,4C,2E,48,FD,75,00,FD,0781
8A28,23,7C,E1,C9,FD,36,00,28,1928
8A30,FD,23,CD,11,8A,FD,77,00,69BA
8A38,FD,23,FE,4C,28,16,FD,36,6D94
8A40,00,2B,FD,23,EB,CD,03,89,13B3
8A48,EB,FD,36,00,2A,FD,23,FD,4E1F
8A50,77,00,FD,23,3E,29,C9,06,26D0
8A56,04,FD,36,00,3F,FD,23,10,39DA
3A60,F8,C9,F5,C5,06,03,1A,FD,5C45
8A68,77,00,13,FD,23,10,F7,FD,367B
8A70,36,00,20,FD,23,C1,F1,C9,13E7
8A78,E6,07,DD,21,A5,8A,32,83,6C97
8A80,8A,DD,7E,00,FE,21,CC,2C,7B40
8A88,8A,FD,77,00,FD,23,C9,B7,72E1
8A90,28,06,13,13,13,3D,20,FA,1606
8A98,CD,62,8A,FD,36,00,20,FD,619D
8AA0,23,C9,4C,44,20,42,43,44,2ECA
8AA8,45,48,4C,21,41,41,44,44,38D0
8AB0,41,44,43,53,55,42,53,42,3F94
8AB8,43,41,4E,44,58,4F,52,4F,3F57
8AC0,52,20,43,50,20,52,4C,43,2CF3
8AC8,52,52,43,52,4C,20,52,52,32D6
8AD0,20,53,4C,41,53,52,41,53,0A51
8AD8,4C,4C,53,52,4C,42,49,54,39EE
8AE0,52,45,53,53,45,54,6E,4F,34FB
8AE8,50,6C,44,20,42,43,2C,40,3A84
8AF0,6C,44,20,28,42,43,29,2C,22E2
8AF8,41,69,4E,43,20,42,43,69,37D7
8B00,4E,43,20,42,64,45,43,20,3572
8B08,42,6C,44,20,42,2C,2A,72,3206
8B10,4C,43,41,65,58,20,41,46,3A34
8B18,2C,41,46,27,61,44,44,20,0E40
8B20,5E,2C,42,43,6C,44,20,41,2A01
8B28,2C,28,42,43,29,64,45,43,1061
8B30,20,42,43,69,4E,43,20,43,0D0F
8B38,64,45,43,20,43,6C,44,20,2A20
8B40,43,2C,2A,72,52,43,41,64,2B9A
8B48,4A,4E,5A,20,2B,6C,44,20,3F80
8B50,44,45,2C,40,6C,44,20,28,30D8
8B58,44,45,29,2C,41,69,4E,43,37D3
8B60,20,44,45,69,4E,43,20,44,0C48
8B68,64,45,43,20,44,6C,44,20,2A18
8B70,44,2C,2A,72,4C,41,6A,52,2882
8B78,20,2B,61,44,44,20,5E,2C,1090
8B80,44,45,6C,44,20,41,2C,28,3AF4
8B88,44,45,29,64,45,43,20,44,3300
8B90,45,69,4E,43,20,45,64,45,35A9
8B98,43,20,45,6C,44,20,45,2C,25E6
8BA0,2A,72,52,41,6A,52,20,4E,05C6
8BA8,5A,2C,2B,6C,44,20,5E,2C,2790
8BB0,40,6C,44,20,28,40,29,2C,31BE
8BB8,5E,69,4E,43,20,5E,69,4E,3854
8BC0,43,20,48,64,45,43,20,48,25EC
8BC8,6C,44,20,48,2C,2A,64,41,26C1
8BD0,41,6A,52,20,5A,2C,2B,61,3017
8BD8,44,44,20,5E,2C,5E,6C,44,3264
8BE0,20,5E,2C,28,40,29,64,45,02A9
8BE8,43,20,5E,69,4E,43,20,4C,27A0
8BF0,64,45,43,20,4C,6C,44,20,2A58
8BF8,4C,2C,2A,63,50,4C,6A,52,2D46
8C00,20,4E,43,2C,2B,6C,44,20,0960
8C08,53,50,2C,40,6C,44,20,28,3E18
8C10,40,29,2C,41,69,4E,43,20,2906
8C18,53,50,69,4E,43,20,21,64,36FE
8C20,45,43,20,21,6C,44,20,21,3641
8C28,2C,2A,73,43,46,6A,52,20,15CC
8C30,43,2C,2B,61,44,44,20,5E,2ADE
8C38,2C,53,50,6C,44,20,41,2C,0C0E
8C40,28,40,29,64,45,43,20,53,0457
8C48,50,69,4E,43,20,41,64,45,3F39
8C50,43,20,41,6C,44,20,41,2C,256E
8C58,2A,63,43,46,72,45,54,20,03CC
8C60,4E,5A,70,4F,50,20,42,43,39B7
8C68,6A,50,20,4E,5A,2C,40,6A,236A
8C70,50,20,40,63,41,4C,4C,20,2DB0
```

Listing 1. Für Sonderheft-Stammleser ein alter Bekannter: Der CPC-Monitor »Smon«

Die Schneider-Sonderhefte von Happy-Computer: eine runde Sache



Nutzen Sie die Bestellmöglichkeit der Schneider-Sonderhefte 1 bis 7 mit der eingehefteten Zahlkarte im vorliegenden Sonderheft von »Happy-Computer«!

```
8C78,4E,5A,2C,40,70,55,53,48,323A
8C80,20,42,43,61,44,44,20,2A,0DAA
8C88,72,53,54,20,30,72,45,54,25D6
8C90,20,5A,72,45,54,6A,50,20,0F18
8C98,5A,2C,40,78,63,41,4C,4C,2B48
8CA0,20,5A,2C,40,63,41,4C,4C,05C8
8CA8,20,40,61,44,43,20,2A,72,0ADE
8CB0,53,54,20,31,72,45,54,20,399C
8CB8,4E,43,70,4F,50,20,44,45,3FFD
8CC0,6A,50,20,4E,43,2C,40,6F,23A7
8CC8,55,54,20,2A,2C,41,63,41,39C3
8CD0,4C,4C,20,4E,43,2C,40,70,37B8
8CD8,55,53,48,20,44,45,73,55,36C7
8CE0,42,20,2A,72,53,54,20,32,28DA
8CE8,72,45,54,20,43,65,58,58,23A4
8CF0,6A,50,20,43,2C,40,69,4E,21CC
8CF8,20,41,2C,2A,63,41,4C,4C,05A8
8D00,20,43,2C,40,78,73,42,43,038B
8D08,20,2A,72,53,54,20,33,72,13C4
8D10,45,54,20,50,4F,70,4F,50,35F6
8D18,20,5E,6A,50,20,50,4F,2C,0F32
8D20,40,65,58,20,28,53,50,29,30C5
8D28,2C,5E,63,41,4C,4C,20,50,0AB0
8D30,4F,2C,40,70,55,53,48,20,20D4
8D38,5E,61,4E,44,20,2A,72,53,3BDF
8D40,54,20,34,72,45,54,20,50,20C8
8D48,45,6A,50,20,28,5E,29,6A,3000
8D50,50,20,50,45,2C,40,65,58,2EA2
8D58,20,44,45,2C,48,4C,63,41,0897
8D60,4C,4C,20,50,45,2C,40,78,3660
8D68,78,4F,52,20,2A,72,53,54,27EA
8D70,20,35,72,45,54,20,50,70,15A0
8D78,4F,50,20,41,46,6A,50,20,3088
8D80,50,2C,40,64,49,63,41,4C,2E4A
8D88,4C,20,50,2C,40,70,55,53,25F9
8D90,48,20,41,46,6F,52,20,2A,221A
8D98,72,53,54,20,36,72,45,54,25E6
8DA0,20,4D,6C,44,20,53,50,2C,0A40
8DA8,5E,6A,50,20,4D,2C,40,65,3FBD
8DB0,49,63,41,4C,4C,20,4D,2C,32F6
8DB8,40,78,63,50,20,2A,72,53,367F
8DC0,54,20,37,69,4E,20,42,2C,202B
8DC8,28,43,29,6F,55,54,20,28,0480
8DD0,43,29,2C,42,73,42,43,20,2856
8DD8,48,4C,2C,42,43,6C,44,20,35A0
8DE0,28,40,29,2C,42,43,6E,45,0065
8DE8,47,72,45,54,4E,69,4D,20,318E
8DF0,30,6C,44,20,49,2C,41,69,0B93
8DF8,4E,20,43,2C,28,43,29,6F,25D1
8E00,55,54,20,28,43,29,2C,43,3BA7
8E08,61,44,43,20,42,43,6C,44,2860
8E10,20,42,43,2C,28,40,29,7B,0A49
8E18,72,45,54,49,7B,6C,44,20,2490
8E20,52,2C,41,69,4E,20,44,2C,2EE4
8E28,28,43,29,6F,55,54,20,28,0480
8E30,43,29,2C,44,73,42,43,20,2836
8E38,48,4C,2C,44,45,6C,44,20,35F0
8E40,28,40,29,2C,44,45,7B,7B,0059
8E48,69,4D,20,31,6C,44,20,41,22A1
8E50,2C,49,69,4E,20,45,2C,28,0DE4
8E58,43,29,6F,55,54,20,28,43,2143
8E60,29,2C,45,61,44,43,20,48,1214
8E68,4C,2C,44,45,6C,44,20,44,23A4
8E70,45,2C,28,40,29,7B,7B,69,28BB
8E78,4D,20,32,6C,44,20,41,2C,2C0E
8E80,52,69,4E,20,48,2C,28,43,3A63
8E88,29,6F,55,54,20,28,43,29,01AF
8E90,2C,48,73,42,43,20,48,4C,0C04
8E98,2C,48,4C,6C,44,20,28,40,09F0
8EA0,29,2C,48,4C,7B,7B,7B,72,10F0
8EA8,52,44,69,4E,20,4C,2C,28,3180
8EB0,43,29,6F,55,54,20,28,43,2143
8EB8,29,2C,4C,61,44,43,20,48,1334
8EC0,4C,2C,48,4C,6C,44,20,48,22B8
8EC8,4C,2C,28,40,29,7B,7B,7B,2C29
8ED0,72,4C,44,7B,7B,73,42,43,27E3
8ED8,20,48,4C,2C,53,50,6C,44,0A04
8EE0,20,28,40,29,2C,53,50,7B,1067
8EE8,7B,7B,7B,69,4E,20,41,2C,28EE
8EF0,28,43,29,6F,55,54,20,28,0480
8EF8,43,29,2C,41,61,44,43,20,28EE
8F00,48,4C,2C,53,50,6C,44,20,3428
8F08,53,50,2C,28,40,29,6C,44,38B8
8F10,49,63,50,49,69,4E,49,6F,305D
8F18,55,54,49,7B,7B,7B,7B,6C,33BE
8F20,44,44,63,50,44,69,4E,44,393C
8F28,6F,55,54,44,7B,7B,7B,7B,2EB9
8F30,6C,44,49,52,63,50,49,52,2998
8F38,69,4E,49,52,6F,54,49,52,29E8
8F40,7B,7B,7B,7B,6C,44,44,52,293A
8F48,63,50,44,52,69,4E,44,52,2ABA
8F50,6F,54,44,52,7B,7B,7B,7B,2D99
8F58,7B,00,00,00,00,00,00,00,3D80
```

Listing. 1. Der CPC-Monitor »Smon« (Schluß)

```
9000,CD,63,96,37,C9,20,20,20,6958
9008,20,20,20,20,20,20,20,20,1FE0
9010,20,20,20,20,20,20,20,20,1FE0
9018,20,20,20,20,20,20,20,31,1FF1
9020,2E,29,20,50,72,6F,67,72,1ED0
9028,61,6D,6D,69,65,72,65,6E,22B4
9030,0A,0D,0A,0D,20,20,20,20,0630
9038,20,20,20,20,20,20,20,20,1FE0
9040,20,20,20,20,20,20,20,20,1FE0
9048,20,20,20,20,20,20,32,2E,1FCA
9050,29,20,4C,65,73,65,6E,0A,118A
9058,0D,0A,0D,20,20,20,20,20,0640
9060,20,20,20,20,20,20,20,20,1FE0
9068,20,20,20,20,20,20,20,20,1FE0
9070,20,20,20,20,20,33,2E,29,1FB9
9078,20,56,65,72,67,6C,65,69,0C2B
9080,63,68,65,6E,0A,0D,0A,0D,21BD
9088,20,20,20,20,20,20,20,20,1FE0
9090,20,20,20,20,20,20,20,20,1FE0
9098,20,20,20,20,20,20,20,20,1FE0
90A0,20,20,34,2E,29,20,48,61,1D59
90A8,75,70,74,6D,65,6E,75,65,2CCF
90B0,0A,0D,0A,0D,20,20,20,20,0630
90B8,20,20,20,20,20,20,20,20,1FE0
90C0,20,20,20,20,20,20,20,20,1FE0
90C8,20,20,20,20,20,20,35,2E,1FC4
90D0,29,20,45,6E,64,65,20,20,1014
90D8,0A,0D,0A,0D,00,20,20,20,0730
90E0,20,20,20,20,20,20,20,2A,1FEA
90E8,2A,2A,2A,2A,2A,20,20,20,19D0
90F0,20,20,45,50,52,4F,4D,20,16B6
90F8,2D,20,50,52,4F,47,52,41,1221
9100,4D,4D,45,52,20,20,56,65,3909
9108,72,73,69,6F,6E,20,31,2E,2DAC
9110,30,20,20,20,20,2A,2A,2A,17D6
9118,2A,2A,0A,0D,0A,0D,20,20,1E14
9120,20,20,20,20,20,20,20,20,1FE0
9128,20,20,20,20,20,20,20,20,1FE0
9130,20,20,20,20,A4,20,31,39,1BFB
9138,38,37,20,62,79,20,50,65,106D
9140,74,65,72,20,42,75,65,6E,2C60
9148,64,67,65,6E,73,20,0A,0D,2281
9150,0A,0D,20,20,20,20,20,20,01A0
9158,20,20,20,20,5F,5F,5F,5F,1D65
9160,5F,5F,5F,5F,5F,5F,5F,5F,3535
9168,5F,5F,5F,5F,5F,5F,5F,5F,3535
9170,5F,5F,5F,5F,5F,5F,5F,5F,3535
9178,5F,5F,5F,5F,5F,5F,5F,5F,3535
9180,5F,5F,5F,5F,5F,5F,5F,5F,3535
9188,5F,5F,5F,5F,5F,5F,5F,5F,3535
9190,0A,0D,0A,0D,0A,0D,00,20,0794
9198,20,20,20,20,20,20,20,20,1FE0
91A0,20,20,20,20,20,20,20,53,1F93
91A8,69,65,20,6B,6F,65,6E,6E,2D2E
91B0,65,6E,20,7A,77,69,73,63,2839
91B8,68,65,6E,20,35,20,54,79,2379
91C0,70,65,6E,20,77,61,65,68,2C1E
91C8,6C,65,6E,21,0A,0D,0A,0D,20ED
91D0,20,20,20,20,20,20,20,20,1FE0
91D8,20,20,20,20,20,20,20,20,1FE0
91E0,20,20,20,20,31,2E,29,20,1F42
91E8,20,44,61,73,20,20,38,2D,0BCD
91F0,4B,42,79,74,65,2D,45,50,3E26
91F8,52,4F,4D,20,20,32,37,36,30F0
9200,34,20,0A,0D,0A,0D,20,20,1394
9208,20,20,20,20,20,20,20,20,1FE0
9210,20,20,20,20,20,20,20,20,1FE0
9218,20,20,32,2E,29,20,20,44,1D6C
9220,61,73,20,31,36,2D,4B,42,2A80
9228,79,74,65,2D,45,50,52,4F,2C73
9230,4D,20,32,37,31,32,38,0A,2A8A
9238,0D,0A,0D,20,20,20,20,20,0640
9240,20,20,20,20,20,20,20,20,1FE0
9248,20,20,20,20,20,20,20,33,1FF3
9250,2E,29,20,20,44,61,73,20,1822
9258,31,36,2D,4B,42,79,74,65,1769
9260,2D,45,50,52,4F,4D,20,32,0BDE
9268,37,31,32,38,41,20,0A,0D,1091
9270,0A,0D,20,20,20,20,20,20,01A0
9278,20,20,20,20,20,20,20,20,1FE0
9280,20,20,20,20,20,20,34,2E,1FC6
9288,29,20,20,44,61,73,20,33,1E77
9290,32,2D,4B,42,79,74,65,2D,1DFF
9298,45,50,52,4F,4D,20,32,37,3A8B
92A0,32,35,36,41,0A,0D,0A,0D,16ED
92A8,20,20,20,20,20,20,20,20,1FE0
92B0,20,20,20,20,20,20,20,20,1FE0
92B8,20,20,20,20,35,2E,29,20,1F62
92C0,20,44,61,73,20,33,32,2D,0B95
92C8,4B,42,79,74,65,2D,45,50,3E26
92D0,52,4F,4D,20,32,37,32,35,307D
92D8,36,0A,0D,0A,0D,20,20,20,1808
92E0,20,20,20,20,20,20,20,20,1FE0
92E8,20,20,20,20,20,20,20,20,1FE0
92F0,20,20,20,20,20,20,20,20,1FE0
92F8,20,20,20,20,20,20,6F,64,1F3A
9300,65,72,20,0A,0D,0A,0D,20,2ADA
9308,20,20,20,20,20,20,20,20,1FE0
9310,20,20,20,20,20,20,20,20,1FE0
9318,20,20,20,36,2E,29,20,20,1ED4
9320,41,75,66,72,75,66,20,64,3434
9328,65,73,20,43,50,43,2D,4D,2DEB
9330,6F,6E,69,74,6F,72,73,20,2416
9338,22,53,4D,4F,4E,22,2E,0A,0A3E
9340,0D,0A,0D,00,0D,0A,20,20,0580
9348,20,20,20,20,20,20,20,20,1FE0
9350,20,20,20,20,20,20,20,20,1FE0
9358,20,20,20,20,20,20,42,69,1F6D
9360,74,74,65,20,77,61,65,68,2B3E
9368,6C,65,6E,20,53,69,65,21,2357
9370,20,00,20,20,20,20,20,20,17E0
9378,20,20,20,20,20,20,20,53,1F93
9380,69,65,20,68,61,62,65,6E,2D64
9388,20,64,65,6E,20,38,2D,4B,02B1
9390,42,79,74,65,2D,54,79,70,372A
9398,20,32,37,36,34,20,61,75,1897
93A0,73,67,65,77,61,65,68,6C,29B0
93A8,74,2E,0A,0D,20,20,20,20,31F0
93B0,20,20,20,20,20,20,20,20,1FE0
93B8,20,5F,5F,5F,5F,5F,5F,5F,0AB5
93C0,5F,5F,5F,5F,5F,5F,5F,5F,3535
93C8,5F,5F,5F,5F,5F,5F,5F,5F,3535
93D0,5F,5F,5F,5F,5F,5F,5F,5F,3535
93D8,5F,5F,5F,5F,5F,5F,5F,5F,3535
93E0,5F,5F,5F,5F,0A,0D,0A,0D,362D
93E8,0A,0D,0A,0D,0A,0D,00,20,0794
93F0,20,20,20,20,20,20,20,20,1FE0
93F8,20,20,20,20,20,53,69,65,1EFB
9400,20,68,61,62,65,6E,20,64,02B4
9408,65,6E,20,31,36,2D,4B,42,2FC0
9410,79,74,65,2D,54,79,70,20,2C74
9418,32,37,31,32,38,20,61,75,1037
9420,73,67,65,77,61,65,68,6C,29B0
9428,74,2E,0A,0D,20,20,20,20,31F0
9430,20,20,20,20,20,20,20,20,1FE0
9438,20,5F,5F,5F,5F,5F,5F,5F,0AB5
9440,5F,5F,5F,5F,5F,5F,5F,5F,3535
9448,5F,5F,5F,5F,5F,5F,5F,5F,3535
9450,5F,5F,5F,5F,5F,5F,5F,5F,3535
9458,5F,5F,5F,5F,5F,5F,5F,5F,3535
9460,5F,5F,5F,5F,5F,5F,0A,0D,35CD
9468,0A,0D,0A,0D,0A,0D,0A,0D,07AD
9470,00,20,20,20,20,20,20,20,0FE0
9478,20,20,20,20,20,20,53,69,1F4F
9480,65,20,68,61,62,65,6E,20,33E8
9488,64,65,6E,20,33,32,2D,4B,25C1
9490,42,79,74,65,2D,54,79,70,372A
9498,20,32,37,32,35,36,41,20,1892
94A0,61,75,73,67,65,77,61,65,2783
94A8,68,6C,74,2E,0A,0D,20,20,2364
94B0,20,20,20,20,20,20,20,20,1FE0
94B8,20,20,20,5F,5F,5F,5F,5F,1A95
94C0,5F,5F,5F,5F,5F,5F,5F,5F,3535
94C8,5F,5F,5F,5F,5F,5F,5F,5F,3535
94D0,5F,5F,5F,5F,5F,5F,5F,5F,3535
94D8,5F,5F,5F,5F,5F,5F,5F,5F,3535
94E0,5F,5F,5F,5F,5F,5F,5F,5F,3535
94E8,0A,0D,0A,0D,0A,0D,0A,0D,07AD
94F0,0A,0D,00,20,20,20,20,20,05A0
94F8,20,20,20,20,20,20,20,20,1FE0
9500,53,69,65,20,68,61,62,65,3F05
9508,6E,20,64,65,6E,20,31,36,3674
9510,2D,4B,42,79,74,65,2D,54,09AA
9518,79,70,20,32,37,31,32,38,2680
9520,41,20,61,75,73,67,65,77,2149
9528,61,65,68,6C,74,2E,0A,0D,2101
9530,20,20,20,20,20,20,20,20,1FE0
9538,20,20,20,20,20,5F,5F,5F,1E9D
9540,5F,5F,5F,5F,5F,5F,5F,5F,3535
9548,5F,5F,5F,5F,5F,5F,5F,5F,3535
9550,5F,5F,5F,5F,5F,5F,5F,5F,3535
9558,5F,5F,5F,5F,5F,5F,5F,5F,3535
9560,5F,5F,5F,5F,5F,5F,5F,5F,3535
9568,5F,5F,0A,0D,0A,0D,0A,0D,39AD
9570,0A,0D,0A,0D,00,20,20,20,0730
9578,20,20,20,20,20,20,20,20,1FE0
9580,20,20,53,69,65,20,68,61,17E9
9588,62,65,6E,20,64,65,6E,20,25C8
9590,33,32,2D,4B,42,79,74,65,1769
9598,2D,54,79,70,20,32,37,32,0A34
95A0,35,36,20,61,75,73,67,65,17DF
95A8,77,61,65,68,6C,74,2E,0A,2B06
95B0,0D,20,20,20,20,20,20,20,0960
95B8,20,20,20,20,20,20,5F,5F,1F61
95C0,5F,5F,5F,5F,5F,5F,5F,5F,3535
95C8,5F,5F,5F,5F,5F,5F,5F,5F,3535
```

Listing 2. Die hexadezimalen Werte von »Eprom« (Eingabe mit dem Programm »CPC«)

```
95D0,5F,5F,5F,5F,5F,5F,5F,5F,3535
95D8,5F,5F,5F,5F,5F,5F,5F,5F,3535
95E0,5F,5F,5F,5F,5F,5F,5F,5F,3535
95E8,5F,5F,5F,0A,0D,0A,0D,0A,3350
95F0,0D,0A,0D,0A,0D,00,80,C6,04AE
95F8,44,80,86,04,00,46,44,00,1310
9600,06,04,0D,0A,52,41,4D,41,004F
9608,4E,46,41,4E,47,20,2D,20,3882
9610,52,41,4D,45,4E,44,45,20,377A
9618,3A,00,0D,0A,45,50,52,4F,1F83
9620,4D,41,4E,46,41,4E,47,20,38FE
9628,2D,20,45,50,52,4F,4D,45,1053
9630,4E,44,45,20,3A,00,0D,0A,3D60
9638,45,50,52,4F,4D,41,44,52,3B86
9640,45,53,53,45,20,3A,00,0D,3995
9648,0A,52,41,4D,41,44,52,45,1E89
9650,53,53,45,20,3A,00,0D,0A,3620
9658,46,45,48,4C,45,52,20,49,3CE9
9660,4E,20,00,CD,FF,BB,3E,02,26BA
9668,CD,0E,BC,21,DD,90,CD,89,752B
9670,96,21,97,91,CD,89,96,21,5DF1
9678,44,93,CD,89,96,CD,28,99,10BD
9680,CD,5A,BB,32,79,97,C3,93,64C1
9688,96,7E,FE,00,CD,5A,BB,23,4D15
9690,20,F7,C9,FE,31,28,19,FE,3AE4
9698,32,28,75,FE,33,28,31,FE,13E4
96A0,34,CA,30,97,FE,35,CA,F0,21B0
96A8,96,FE,36,20,CA,C3,7B,97,757D
96B0,3E,02,CD,0E,BC,21,DD,90,028E
96B8,CD,89,96,21,72,93,CD,89,54DF
96C0,96,21,05,90,CD,89,96,21,4FA1
96C8,F6,95,CD,60,97,C3,50,97,4663
96D0,3E,02,CD,0E,BC,21,DD,90,028E
96D8,CD,89,96,21,F3,94,CD,89,50CB
96E0,96,21,05,90,CD,89,96,21,4FA1
96E8,F9,95,CD,60,97,C3,50,97,41E3
96F0,3E,02,CD,0E,BC,21,DD,90,028E
96F8,CD,89,96,21,75,95,CD,89,54FF
9700,96,21,05,90,CD,89,96,21,4FA1
9708,FC,95,CD,60,97,C3,50,97,4363
9710,3E,02,CD,0E,BC,21,DD,90,028E
9718,CD,89,96,21,F0,93,CD,89,50CF
9720,96,21,05,90,CD,89,96,21,4FA1
9728,F6,95,CD,60,97,C3,50,97,4663
9730,3E,02,CD,0E,BC,21,DD,90,028E
9738,CD,89,96,21,71,94,CD,89,54DB
9740,96,21,05,90,CD,89,96,21,4FA1
9748,FF,95,CD,60,97,C3,50,97,42E3
9750,F5,21,44,93,CD,89,96,F1,76E1
9758,CD,28,99,CD,5A,BB,18,63,731F
9760,11,6C,97,06,03,7E,12,23,00E7
9768,13,10,FA,C9,00,00,00,00,1E50
9770,00,00,00,00,00,00,00,00,0000
9778,00,00,00,3A,00,80,FE,31,006D
9780,28,07,21,8C,97,CD,89,96,1F28
9788,C9,C3,00,80,0C,42,69,74,5D8E
9790,74,65,20,4D,6F,6E,69,74,21F6
9798,6F,72,20,6E,61,63,68,6C,2BD8
97A0,61,64,65,6E,20,75,6E,64,23AC
97A8,20,45,70,72,6F,6D,6D,65,0A13
97B0,72,20,6E,65,75,20,73,74,392A
97B8,61,72,74,65,6E,00,D1,F1,26F3
97C0,C3,50,97,FE,30,CA,50,97,6A1F
97C8,FE,34,CA,63,96,FE,35,C8,6A9A
97D0,D2,50,97,F5,FE,31,20,05,67C1
97D8,21,02,96,18,03,21,1A,96,037E
97E0,CD,89,96,CD,78,99,D2,BF,5A6F
97E8,97,B7,ED,52,23,22,71,97,7F25
97F0,22,73,97,F1,F5,D5,FE,31,1501
97F8,20,05,21,36,96,18,03,21,12F7
9800,47,96,CD,89,96,CD,7D,99,10D7
9808,30,B4,3A,79,97,FE,34,38,32C0
9810,22,D1,F1,FE,31,F5,D5,20,3756
9818,08,CB,74,28,16,CB,B4,18,382C
9820,06,CB,72,28,0E,CB,B2,E5,3FDD
9828,21,6C,97,06,03,CB,FE,23,1BEB
9830,10,FB,E1,D1,EB,06,F8,F1,21B1
9838,FE,33,CA,AB,99,FE,32,CA,67AE
9840,AF,99,FE,31,C2,50,97,EB,6B85
9848,3E,05,32,6F,97,22,75,97,1ABD
9850,ED,53,77,97,2A,71,97,7D,6417
9858,B4,CA,ED,99,11,00,01,ED,7CD7
9860,52,7B,22,71,97,30,07,15,30F3
9868,ED,53,71,97,24,7D,ED,47,6459
9870,B7,20,01,EB,22,73,97,2A,5CC8
9878,75,97,ED,5B,77,97,CB,F4,07D6
9880,CD,D1,99,0D,3E,FB,CD,14,42A2
9888,99,CD,09,BB,30,05,FE,FC,75C4
9890,CA,ED,99,D5,3A,6D,97,F5,418F
9898,CD,05,99,1A,0D,ED,79,0E,7660
98A0,E2,3C,28,06,3A,6E,97,ED,7ACB
98A8,79,76,F1,ED,79,23,13,ED,327F
98B0,57,3D,ED,47,20,DE,3D,ED,3FFF
98B8,47,CD,14,99,2A,75,97,D1,1AAB
98C0,CD,E3,99,CA,48,98,3A,6F,41FB
98C8,97,3D,32,6F,97,CA,B9,99,420B
98D0,E5,D5,2A,73,97,ED,5B,71,427B
98D8,97,19,22,71,97,D1,E1,C3,486D
98E0,4D,98,3A,6C,97,CD,05,99,061F
98E8,0D,ED,48,ED,57,B7,79,28,3AAE
98F0,05,1A,B9,37,C0,3F,12,23,16AB
98F8,13,E5,2A,73,97,2B,7D,B4,36EA
9900,22,73,97,E1,C9,0E,E1,ED,166F
9908,61,03,3C,ED,79,3D,ED,79,3B8F
9910,0B,ED,69,C9,E5,2A,6D,97,38BD
9918,A5,ED,79,EE,04,ED,79,E1,6B87
9920,06,B1,76,10,FD,06,F8,C9,2649
9928,CD,8A,BB,CD,06,BB,F5,CD,5C4B
9930,8D,BB,F1,C9,CD,28,99,FE,7DF4
9938,61,D8,FE,7B,D0,D6,20,C9,1BA1
9940,CD,34,99,FE,FC,C8,FE,0D,7271
9948,C8,D6,30,38,F3,FE,0A,38,504C
9950,0B,D6,07,FE,10,30,E9,FE,3E6C
9958,0A,3F,30,E4,F5,CD,9D,99,07BF
9960,F1,C9,CD,87,99,CD,40,99,5CF5
9968,D0,C5,07,07,07,07,4F,CD,59A7
9970,40,99,30,02,B1,37,C1,C9,047F
9978,CD,7D,99,D0,EB,CD,62,99,63D1
9980,D0,67,CD,65,99,6F,C9,F5,6A23
9988,3E,20,CD,5A,BB,F1,C9,7C,0CF2
9990,CD,94,99,7D,F5,0F,0F,0F,50F5
9998,0F,CD,9D,99,F1,E6,0F,FE,2A00
99A0,0A,38,02,C6,07,C6,30,CD,04AD
99A8,5A,BB,C9,3E,01,18,01,AF,19C5
99B0,ED,47,CB,B4,CD,E3,99,30,7186
99B8,0F,22,6F,97,21,56,96,CD,0A21
99C0,89,96,2A,6F,97,CD,8F,99,64BB
99C8,3A,6C,97,21,00,40,CD,05,166F
99D0,99,3E,80,18,02,3E,90,0E,5346
99D8,E3,18,05,3A,6C,97,0E,E2,7542
99E0,ED,79,C9,CD,D5,99,CD,E2,7884
99E8,98,D8,20,FA,C9,CD,C8,99,75D5
99F0,C3,50,97,FF,00,FF,00,FF,6B93
```

Listing 2. »Eprom« (Schluß)

Pforten zur Hardwarewelt

Mit den Assembler-Befehlen OUT und IN sowie den Basic-Befehlen OUT und INP sind Sie in der Lage, die Hardware Ihres CPC direkt anzusprechen und damit sämtliche Komponenten des Computers unmittelbar zu steuern (Daten schreiben) und zu kontrollieren (Daten lesen).

Hardware-Freaks verschmähen beim Programmieren den Umweg über das Betriebssystem, wenn sich einzelne Bausteine im Computer direkt ansprechen lassen. Und Maschinensprache-Programmierer wissen ohnehin, daß die direkte Programmierung über OUT- und IN-Befehle viel höhere Geschwindigkeiten zuläßt als der Aufruf der zugehörigen Betriebssystem-Routinen über die Vektoren des CPC.

Oft ist die unmittelbare Programmierung der Hardware auch der einzige Weg, um eine gewünschte Funktion zu verwirklichen. Man denke nur an die Bild- und Toneffekte, die einige kommerzielle Spielprogramme erzeugen, indem sie direkt den Video-Controller und den Tongenerator des CPC programmieren.

Der CPC enthält an wichtigen Hardware-Komponenten den Prozessor Z-80, der die restliche Hardware steuert, das Gate Array, den Video-Controller 6845, den Disketten-Controller 765, den Parallel-Portbaustein 8255 und den Tongenerator 8912.

Das Programmieren dieser Bausteine ist einfacher, als viele Computerbesitzer denken. Der Speicher eines Computers läßt sich verwalten, indem man mit dem Assembler-Befehl LD oder den Basic-Befehlen POKE und PEEK Daten in den Speicher schreibt beziehungsweise aus dem Speicher liest. Damit der Computer weiß, in welche Speicherposition er schreiben oder aus welcher Speicherposition er lesen soll, müssen Sie bei jeder Operation eine Speicheradresse angeben.

Das Programmieren der Computerbausteine erfolgt nach dem gleichen Prinzip. Auch hier lassen sich Daten schreiben und lesen, indem man die Assembler-Befehle OUT und IN beziehungsweise die Basic-Befehle OUT und INP verwendet. Nur muß man in diesem Fall zur Adressierung nicht eine Speicheradresse, sondern eine sogenannte Portadresse angeben. Diese Portadresse bestimmt, welcher Baustein angesprochen ist. Die Portadressen der einzelnen Bausteine sind durch die interne Verdrahtung des Computers bereits festgelegt.

Benutzen Sie beispielsweise in Basic den Befehl »OUT &EF00,0«, so wird der Wert 0 auf den Druckeranschluß gelegt. Umgekehrt können Sie mit dem INP-Befehl den Zustand eines Bausteins über eine Portadresse erfragen. Da das Betriebssystem des Schneider CPC jedoch die

Die Portadressen des Schneider CPC

Hex.	Binär	Baustein
7Fxx	0111 1111 xxxx xxxx	Gate Array
BCxx	1011 1100 xxxx xxxx	Video-Controller 6845 (Adreßregister zur Auswahl eines Datenregisters)
BDxx	1011 1101 xxxx xxxx	Video-Controller 6845 (Datenregister, schreiben)
BFxx	1011 1111 xxxx xxxx	Video-Controller 6845 (Datenregister, lesen)
DFxx	1101 1111 xxxx xxxx	Nummer des ausgewählten Erweiterungs-ROM
EFxx	1110 1111 xxxx xxxx	Druckeranschluß
F4xx	1111 0100 xxxx xxxx	Parallel-Portbaustein 8255 (Kanal A)
F5xx	1111 0101 xxxx xxxx	Parallel-Portbaustein 8255 (Kanal B)
F6xx	1111 0110 xxxx xxxx	Parallel-Portbaustein 8255 (Kanal C)
F7xx	1111 0111 xxxx xxxx	Parallel-Portbaustein 8255 (Steuerregister)
FA7E	1111 1010 0111 1110	Laufwerks-Flipflop (Flipflop = 1: Internes ROM des CPC deaktiviert)
FB7E	1111 1011 0111 1110	Disketten-Controller 765 (Statusregister)
FB7F	1111 1011 0111 1111	Disketten-Controller 765 (Datenregister)

x: Beliebiger Bit-Wert

Tabelle 1. Die wichtigsten Portadressen des Schneider CPC auf einen Blick

Die Register des Video-Controllers 6845 (BCxx, BDxx, BFxx)

Register	Bit	Funktion
0	0 bis 7	Zahl der (teilweise unsichtbaren) Zeichen pro Zeile -1
1	0 bis 7	Zahl der tatsächlich angezeigten Zeichen pro Zeile
2	0 bis 7	Rasterzeilen-Position des HSYNC-Signals in CLK-Takten (bestimmt horizontale Verschiebung)
3	0 bis 3	Länge des HSYNC-Impulses in CLK-Takten
4	0 bis 6	Zahl der (teilweise unsichtbaren) Zeichenzeilen -1
5	0 bis 4	Feineinstellung für Register 4 in Rasterzeilen
6	0 bis 6	Zahl der tatsächlich angezeigten Zeichenzeilen
7	0 bis 6	Zeichenposition des VSYNC-Signals (bestimmt senkrechte Verschiebung)
8	0 und 1	Zeilensprungmodus (Interlace) ein/aus
9	0 bis 4	Höhe eines Zeichens in Rasterzeilen
10	0 bis 7	Erste Rasterzeile des Cursors
11	0 bis 7	Letzte Rasterzeile des Cursors
12	0 bis 7	Bit 0 bis Bit 7 der Startadresse des Bildschirmspeichers
13	0 bis 5	Bit 8 bis 14 der Startadresse des Bildschirmspeichers
14	0 bis 7	Bit 0 bis Bit 7 der Adresse des Cursors
15	0 bis 5	Bit 8 bis Bit 14 der Adresse des Cursors
16	0 bis 7	Bit 0 bis Bit 7 der Bildschirmadresse, bei der ein Lichtgriffelimpuls auftrat
17	0 bis 5	Bit 8 bis Bit 14 der Bildschirmadresse, bei der ein Lichtgriffelimpuls auftrat

Nur die Register 14 bis 17 können zusätzlich gelesen werden.

Tabelle 3. Der umfangreiche Registersatz des Video-Controllers 8645

Die Betriebsarten des Tongenerators 8912

BC1	BDIR	Betriebsart
0	0	Baustein gesperrt
1	1	Adressierung eines gewünschten Registers über DA0 bis DA7
0	1	Eingabe eines Wertes in ein zuvor ausgewähltes Register über DA0 bis DA7
1	0	Auslesen eines Wertes aus einem zuvor gewählten Register über DA0 bis DA7

Tabelle 5. Vier Betriebsarten kennt der Tongenerator des Schneider CPC

meisten Portzustände der Bausteine gleich nach Ausführung des OUT- oder INP-Befehls automatisch zurückgesetzt, läßt sich unter Basic die gewünschte Wirkung oft nicht erreichen. Besser ist es deshalb, die Bausteine in Maschinencode beziehungsweise in Assemblersprache zu programmieren und auszulesen, weil man nur auf dieser Ebene volle Kontrolle über die Hardware des Computers erlangen kann.

Die Programmierung des Gate Array (7Fxx)

Bit	Funktion
Multifunktionsregister	
0	Bildschirmmodus
1	Bildschirmmodus Bit 1=0 und Bit 0=0: Modus 0 mit Blinken Bit 1=0 und Bit 0=1: Modus 1 Bit 1=1 und Bit 0=0: Modus 2 Bit 1=1 und Bit 0=1: Modus 0 ohne Blinken
2	Unteres ROM (0000 bis 3FFF hex) ausschalten (Bit 2=1)
3	Oberes ROM (C000 bis FFFF hex) ausschalten (Bit 3=1)
4	Interrupt-Zähler löschen
5	Nicht benutzt
6	Registerauswahl
7	Registerauswahl Bit 7=0 und Bit 6=0: Bit 0 bis 5 in Farbnummer-Register (PEN) schreiben Bit 7=0 und Bit 6=1: Bit 0 bis 5 in Farbwert-Register (INK) schreiben Bit 7=1 und Bit 6=0: Bit 0 bis 5 in Multifunktions-Register schreiben Bit 7=1 und Bit 6=1: Bit 0 bis 2 zur RAM-Konfiguration verwenden (nur CPC 6128)

Tabelle 2. Auf diese Weise läßt sich das Gate Array programmieren

Da der Z80-Prozessor nur bei den Befehlen »OUT (C),Register« und »IN Register,(C)« die Ein- und Ausgabe von 16-Bit-Portadressen erlaubt, dürfen beim CPC ausschließlich diese Befehle zur Adressierung der Ports verwendet werden. Der Inhalt des B-Registers bildet jeweils die obere Hälfte der Adresse, und die untere, meist nicht relevante Hälfte, steht in C.

Eine typische Befehlsfolge zur Programmierung eines Bausteins sieht folgendermaßen aus:

```
LD  A,H89   ;Wert laden, der in
             das Gate Array
             geschrieben werden soll
LD  B,H7F   ;Obere Hälfte der
             Portadresse laden
OUT (C),A   ;Wert in das Gate
             Array schreiben
```

Das Ganze läßt sich noch kürzer und eleganter formulieren, indem man 16-Bit-Register verwendet.

```
LD  BC,H7F89 ;Wert und Adresse laden
OUT (C),C    ;Wert in das Gate Array
              schreiben
```

Das Einlesen von Betriebszuständen funktioniert auf die gleiche Art und Weise, sofern die Hardwarekonfiguration des CPC das Lesen aus den Portadressen zuläßt. (So läßt sich beispielsweise ein Wert vom Druckeranschluß einlesen, doch dieser Wert

Nach wie vor:

Unschl

Spitzen-Software von Markt&T

MicroPro. ASHTON·TATE MICROSO

WordStar, dBASE II, MULT

WordStar 3.0 mit MailMerge
Ein Bestseller unter den Textverarbeitungsprogrammen, der Ihnen bildschirmorientierte Formatierung, deutschen Zeichensatz und DIN-Tastatur sowie integrierte Hilfstexte bietet. Mit MailMerge können Sie Serienbriefe mit persönlicher Anrede an eine beliebige Anzahl von Adressen schreiben und auch die Adreßaufkleber ausdrucken.

dBASE II, Version 2.41
dBASE II, das meistverkaufte Programm unter den Datenbanksystemen, eröffnet Ihnen optimale Möglichkeiten der Daten- und Dateihandhabung. Einfach und schnell können Datenstrukturen definiert, benutzt und geändert werden. Der Datenzugriff erfolgt sequentiell oder nach frei wählbaren Kriterien, die integrierte Kommandosprache ermöglicht den Aufbau kompletter Anwendungen wie Finanzbuchhaltung, Lagerverwaltung, Betriebsabrechnung usw.

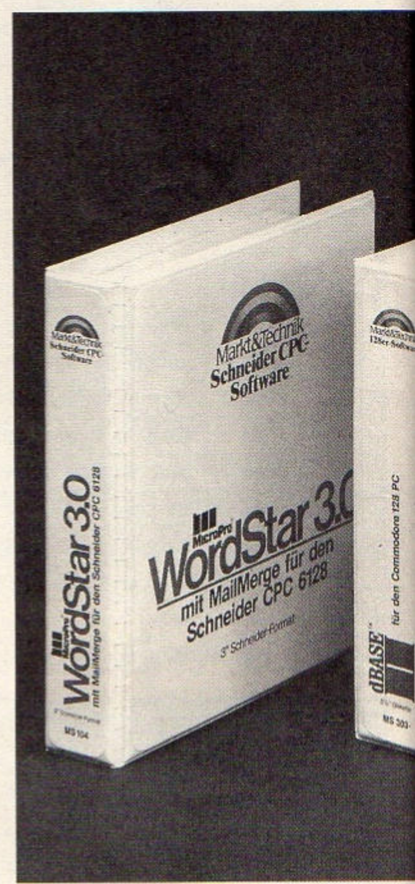
MULTIPLAN, Version 1.06
Wenn Sie die zeitraubende manuelle Verwaltung tabellarischer Aufstellungen mit Bleistift, Radiergummi und Rechenmaschine satt haben, dann ist MULTIPLAN, das System zur Bearbeitung »elektronischer Datenblätter«, genau das Richtige für Sie! Das benutzerfreundliche und leistungsfähige Tabellenkalkulationsprogramm kann bei allen Analyse- und Planungsberechnungen eingesetzt werden.

Sie erhalten jedes **WordStar-, dBASE II- und MULTIPLAN-Programm** für Ihren Schneider-Computer oder Commodore 128PC fertig angepaßt (Bildschirmsteuerung). Jeweils Originalprodukte! Jedes Programmpaket enthält außerdem ein ausführliches Handbuch mit kompakter Befehlsübersicht.

Version	Format	Bestell-Nummer		
		WordStar	dBASE	Multiplan
Schneider CPC 464*/664*	3"	50101	50301	50201
Schneider CPC 464*/664*	5¼"	50102	50302	50202
Schneider CPC 6128	3"	50104	50304	50204
Schneider Joyce	3"	50105	50305	50205
Commodore 128	5¼"	50103	50303	50203

*dBASE II und MULTIPLAN für die Schneider CPC 464/664 sind nur lauffähig mit einer Speichererweiterung auf mindestens 128 Kbyte und einer CP/M-Version für 62 Kbyte.

Für Atari ST
WordStar 3.0 (50106, DM 199,–*), dBASE II (50306, DM 348,–*)



Diese Markt & Technik-Softwareprodukte erhalte
in den Computer-Abteilungen der Kaufhäuser, bei Ihrem Computerhä
im Buchhandel oder direkt beim Verlag gegen Vorausk

702201

agbar!

nik

LAN - für CP/M Computer

Jedes Buch kostet
DM 49,–
(sFr. 45,10/öS 382,20)
Erhältlich bei Ihrem Buchhändler.

Und dazu die weiterführende Literatur:
WordStar für den Schneider CPC
Best.-Nr. 90180, ISBN 3-89090-180-8
WordStar für den Commodore 128PC
Best.-Nr. 90181, ISBN 3-89090-181-6
dBASE II für den Commodore 128PC
Best.-Nr. 90189, ISBN 3-89090-189-1
dBASE II für den Schneider CPC
Best.-Nr. 90188, ISBN 3-89090-188-3
MULTIPLAN für den Schneider CPC
Best.-Nr. MT 835, ISBN 3-89090-186-7
MULTIPLAN für den Commodore 128PC
Best.-Nr. MT 836, ISBN 3-89090-189-1

Hardware-Anforderungen für Schneider-Computer:
Schneider CPC 464, CPC 664, CPC 6128, Joyce, beliebiger Drucker mit Centronics-Schnittstelle.
Hardware-Anforderungen für Commodore 128PC:
Commodore 128/128D, Diskettenlaufwerk, 80-Zeichen-Monitor, Commodore-Drucker oder Drucker mit Centronics-Schnittstelle (ohne zwischengeschaltetes Interface).
Übrigens gibt es WordStar, dBASE und MULTIPLAN auch für NDR-Computer. Zu beziehen bei Graf Elektronik Systeme GmbH, Magnusstraße 13, 8960 Kempten.

Zeitschriften · Bücher
Software · Schulung
arkt&Technik Verlag Aktiengesellschaft
s-Pinsel-Str. 2, 8013 Haar bei München

Bestellungen im Ausland bitte an untenstehende Adressen.
Schweiz: Markt & Technik Vertriebs AG, Kollerstr. 3, CH-6300 Zug, Tel. (042) 415656
Österreich: Ueberreuter Media Handels- und Verlagsges. mbH., Alser Str. 24, A-1091 Wien, Tel. (0222) 481538-0

Der Parallel-Portbaustein 8255

Bit	Funktion
Kanal A (F4xx)	
0	DA0 für Tongenerator 8912
1	DA1 für Tongenerator 8912
2	DA2 für Tongenerator 8912
3	DA3 für Tongenerator 8912
4	DA4 für Tongenerator 8912
5	DA5 für Tongenerator 8912
6	DA6 für Tongenerator 8912
7	DA7 für Tongenerator 8912
Kanal B (F5xx)	
0	VSYNC (vertikale Video-Synchronisation)
1	Lötbrücke LK1 für Herstellercode
2	Lötbrücke LK2 für Herstellercode
3	Lötbrücke LK3 für Herstellercode
4	Lötbrücke LK4 für Herstellercode
5	EXP (Erweiterungseinheit übernimmt Kontrolle)
6	BUSY (Drucker beschäftigt)
7	RD DATA (Daten vom Kassettenrecorder)
Kanal C (F6xx)	
0	D0 für Tastaturdekoder
1	D1 für Tastaturdekoder
2	D2 für Tastaturdekoder
3	D3 für Tastaturdekoder
4	MOTOR (Kassettenrecoder-Motor einschalten)
5	WR DATA (Daten für Kassettenrecoder)
6	BC1 (zur Auswahl der Tongenerator-Betriebsart)
7	BDIR (zur Auswahl der Tongenerator-Betriebsart)
Steuerregister (F7xx)	
0	Bit 0=0: Kanal C (Bit 0 bis Bit 3) ist Ausgabekanal Bit 0=1: Kanal C (Bit 0 bis Bit 3) ist Eingabekanal
1	Bit 1=0: Kanal B ist Ausgabekanal
2	Bit 1=1: Kanal B ist Eingabekanal Bit 2=0: Modus 0 Bit 2=1: Modus 1
3	Bit 3=0: Kanal C (Bit 4 bis Bit 7) ist Ausgabekanal Bit 3=1: Kanal C (Bit 4 bis Bit 7) ist Eingabekanal
4	Bit 4=0: Kanal A ist Ausgabekanal Bit 4=1: Kanal A ist Eingabekanal
5	Bit 5=0 und Bit 6=0: Modus 0 Bit 5=1 und Bit 6=0: Modus 1
6	Bit 6=0: Modus 0 oder 1 Bit 6=1: Modus 2
7	Bit 7=1

Tabelle 4. Der Parallel-Portbaustein 8255 ist gleich für mehrere Aufgaben verantwortlich

Die Register des Tongenerators 8912

Register	Bit	Funktion
0	0 bis 7	Bit 0 bis Bit 7 des Frequenzteilers für Kanal A
1	0 bis 3	Bit 8 bis 11 des Frequenzteilers für Kanal A
2	0 bis 7	Bit 0 bis Bit 7 des Frequenzteilers für Kanal B
3	0 bis 3	Bit 8 bis 11 des Frequenzteilers für Kanal B
4	0 bis 7	Bit 0 bis Bit 7 des Frequenzteilers für Kanal C
5	0 bis 3	Bit 8 bis 11 des Frequenzteilers für Kanal C
6	0 bis 4	Frequenzkontrolle des Rauschgenerators
7	0 1 2 3 4 5 6 7	Bit 0=0: Ton-Freigabe für Kanal A Bit 1=0: Ton-Freigabe für Kanal B Bit 2=0: Ton-Freigabe für Kanal C Bit 3=0: Rausch-Freigabe für Kanal A Bit 4=0: Rausch-Freigabe für Kanal B Bit 5=0: Rausch-Freigabe für Kanal C Bit 6=0: Freigabe für Datenkanal A Nicht benutzt
8	0 bis 3 4	Lautstärke für Kanal A Bit 4=1: Hüllkurve bestimmt Lautstärke für Kanal A
9	0 bis 3 4	Lautstärke für Kanal B Bit 4=1: Hüllkurve bestimmt Lautstärke für Kanal B
10	0 bis 3 4	Lautstärke für Kanal C Bit 4=1: Hüllkurve bestimmt Lautstärke für Kanal C
11	0 bis 7	Bit 0 bis 7 des Hüllkurven-Frequenzteilers
12	0 bis 7	Bit 8 bis 15 des Hüllkurven-Frequenzteilers
13	0 1 2 3	Bit 0=1: Hüllkurve konstant halten Bit 1=1: Richtung periodisch wechseln Bit 2=0: Absteigende Hüllkurve Bit 2=1: Aufsteigende Hüllkurve Bit 3=1: Hüllkurve fortführen
14	0 bis 7	Bit 0 bis 7 des Datenkanals A

Tabelle 6. Durch die Programmierung des Tongenerators 8912 lassen sich neue akustische Effekte erzielen

beträgt immer 255, weil der Druckeranschluß nur als Datenausgabe funktioniert.)

```
LD  B,FB   ;Obere Hälfte der
            Portadresse laden
LD  C,7E   ;Untere Hälfte der
            Portadresse laden
IN  A,(C)  ;Wert aus dem Status-
            register des Disketten-
            Controllers lesen
```

oder entsprechend kürzer

```
LD BC,FB7E ;Portadresse laden
IN A,(C)   ;Wert aus dem Status-
            register des Disketten-
            Controllers lesen
```

Tabelle 1 zeigt die wichtigsten Portadressen des Schneider CPC. Eine besondere Rolle spielt das Gate Array, das einen Großteil der Funktionen im CPC steuert, jedoch nur über eine einzige Portadresse verfügt. Über diese Portadresse lassen sich aber drei (beim CPC 6128 vier) Register ansprechen (Tabelle 2).

Der Video-Controller 6845 ist für den Aufbau des Bildschirminhalts zuständig. Tabelle 3 zeigt den Registersatz dieses Bausteins, der dem erfahrenen Programmierer viele Wege zur Bildgestaltung (zum Beispiel ein anderes Bildformat, eine andere Bildfrequenz oder die Fähigkeit zur Programmierung von Rasterinterrupts) bietet.

Der Parallel-Portbaustein 8255 ist für Tastatur- und Joystickabfrage, Programmierung des Tongenerators, Steuerung des Kassettenrecorders und des Druckers zuständig. Tabelle 4 zeigt die Funktionen der einzelnen Bits aller drei Kanäle und die Bedeutung des Steuerregisters.

Wenn der Modus 0 des 8255 ausgewählt ist, erfolgt die Ein- und Ausgabe von Daten unidirektional ohne Handshake-Signale. Befindet sich der 8255 dagegen im Modus 1, so werden zusätzlich Handshake-Signale erzeugt, und arbeitet der 8255 im Modus 2, so erfolgt die Datenein- und -ausgabe (jedoch nur für Kanal A) bidirektional im Handshake-Betrieb. Der CPC nutzt jedoch ausschließlich den Modus 0.

Wer über den Portbaustein 8255 den Tongenerator 8912 programmieren möchte, der kann sich in Tabelle 5 über die Betriebsarten des Bausteins und in Tabelle 6 über den Registersatz informieren.

Auf die Programmierung des Disketten-Controllers 765 wird an dieser Stelle nicht eingegangen, da eine Beschreibung der komplexen Fähigkeiten des 765-Bausteins den Rahmen dieses Beitrags sprengen würde.

(Thomas Bullinger/ma)

Die Diskette – (k)ein Buch mit sieben Siegeln

In der Zwischenzeit ist die Diskette wohl das beliebteste Speichermedium – nicht nur für Schneider-Computer. Dazu haben nicht zuletzt die CPCs 664 und 6128 durch ihre von vornherein eingebauten Disketten-Laufwerke beigetragen. Die Dokumentation in den mitgelieferten Handbüchern läßt jedoch viele Fragen offen. So haben wir die fehlende Information zusammengetragen.

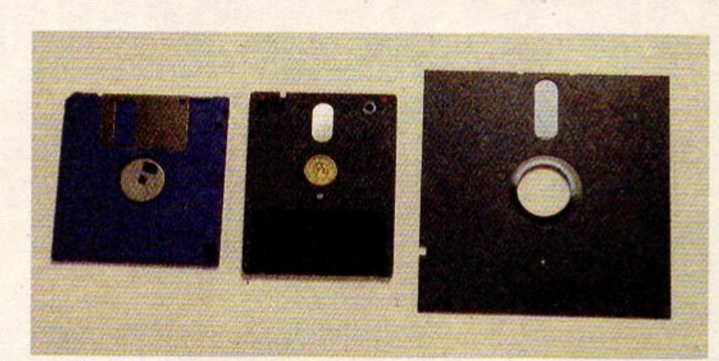


Bild 1. Die drei für CPs erhältlichen Diskettenformate: 3½ Zoll, 3 Zoll, und 5¼ Zoll

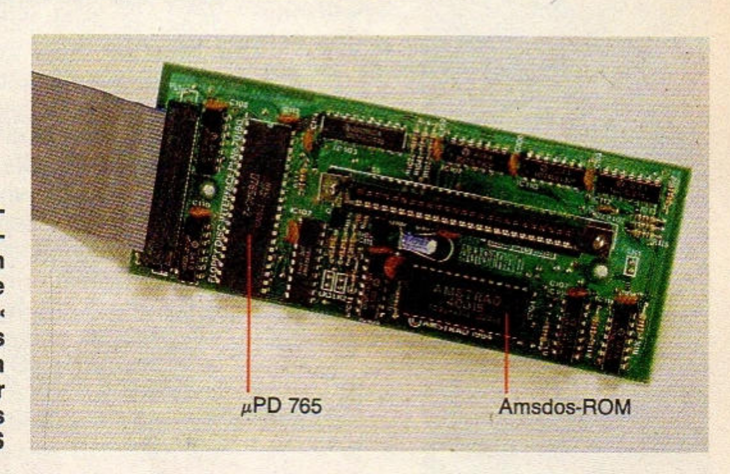


Bild 2. Im geöffneten Controllergehäuse erkennt man auf der Platine deutlich »Herz« und »Gehirn« des Controllers: den Mikroprozessor 765 (FDC) und das ROM mit dem DOS

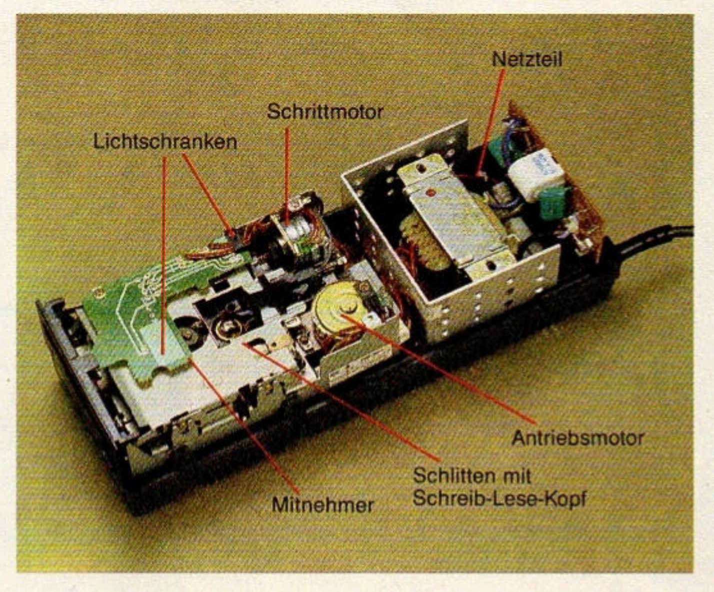


Bild 3. Das geöffnete Laufwerksgehäuse einer DDI-1 offenbart das Innenleben

Die Geschwindigkeit, mit der die Diskette ihre Aufgabe erledigt, wird nur noch vom Komfort ihrer Handhabung überboten. In ihrem Preis-Leistungs-Verhältnis schlägt die Diskette jedes andere Speichermedium. Aber die Entwickler bei Amstrad taten einen Fehlgriff, als Sie entgegen allen anderen Computerherstellern, das 3-Zoll- dem 3½-Zoll-Format vorzogen. Da Schneider und ihr englischer Partner Amstrad also die einzigen Anbieter dieses Formats sind, hat sich der recht hohe Diskettenpreis leider nicht wesentlich nach unten entwickelt. Mit immer noch zirka acht Mark pro Diskette sind diese Datenträger zwar im Preis gesunken, den Preis einer billigen 5¼-Zoll-Diskette von zum Teil nur noch ein bis zwei Mark werden sie aber nie erreichen. Diesen Preisvorteil des »älteren« Formats machten sich Firmen wie Vortex und Cumana zunutze, indem sie schon bald 5¼-Zoll-Laufwerke für die CPC-Serie anboten. Vortex bietet alternativ auch Laufwerke im 3½-Zoll-Format an. Disketten mit diesem Maß gehen mit den Vorteilen der beiden anderen Systeme einen guten Kompromiß ein. Sie liegen preislich ziemlich genau in der Mitte zwischen den beiden konkurrierenden Größen, verfügen (mit den entsprechenden Laufwerken) über dieselbe hohe Speicherkapazität wie die 5¼-Zoll-Disketten, bieten aber, ähnlich ihrem 3½-Zoll-Pendant, den besseren Schutz der empfindlichen Diskettenoberfläche vor mechanischer Unbill. Bild 1 zeigt je einen typischen Vertreter der drei Diskettentypen. Mittlerweile besitzt also ein großer Teil der Schneider-Benutzer ein Diskettenlaufwerk zu seinem Computer. Leider weiß aber noch längst nicht jeder, was in dem »surrenden Kasten« genau vor sich geht. Diesem Umstand wollen wir mit unserem Beitrag Abhilfe schaffen. Gehören Sie schon zu den »alten Hasen«, denen man in bezug auf die normale Diskettenhandhabung nichts mehr vormachen kann, sollten Sie auf Seite 38 den weiterführenden Beitrag studieren.

Die Diskettenstation besteht im wesentlichen aus Mechanik (dem Laufwerk), Elektronik (dem Controller) und Software, dem Disc-Operating-System – kurz »DOS« genannt (Bild 2). Das Gehäuse des Schneider 3-Zoll-Laufwerks enthält neben dem Lauf-

Die neue Happy-Computer im Juli

CPC und Grafik
Informieren Sie sich in dieser Ausgabe von Happy-Computer über die großartigen graphischen Fähigkeiten des Schneiders CPC.

Desktop Publishing
Die wichtigsten DTP-Programme haben wir für Sie getestet. Fortgeschrittene sollten sich den Programmierwettbewerb vormerken, den wir gemeinsam mit Atari veranstalten.

Bitgedrängel
Wir beleuchten die Speicherlandschaft der nächsten Jahrzehnte, untersuchen günstige Harddisks und stellen ein CD-Rom vor, auf dem mehrere hundert MByte Public Domain-Software gespeichert sind.

Schachcomputer de Luxe
Lesen Sie unseren interessanten Erfahrungsbericht über den sensationellen »Mephisto Dallas 68020«.

C64: Listing
Mit unserem Listing »Happy-Packer« können Programme für den C64 im Durchschnitt auf weniger als 70 Prozent ihrer ursprünglichen Länge gekürzt werden, ohne daß ein Bit verlorengeht.

Ab 26.6.1987 im Zeitschriftenhandel

Gutschein

FÜR EIN KOSTENLOSES PROBEEXEMPLAR VON HAPPY-COMPUTER

JA, ich möchte »Happy-Computer« kennenlernen.
Senden Sie mir bitte die aktuellste Ausgabe kostenlos als Probeexemplar. Wenn mir »Happy-Computer« gefällt und ich es regelmäßig weiterbeziehen möchte, brauche ich nichts zu tun: Ich erhalte »Happy-Computer« dann regelmäßig frei Haus per Post und bezahle pro Jahr nur DM 66,— statt DM 72,— Einzelverkaufspreis (Ausland auf Anfrage).

Vorname, Name

Straße PLZ, Ort

Datum 1. Unterschrift

Mir ist bekannt, daß ich diese Bestellung innerhalb von 8 Tagen bei der Bestelladresse widerrufen kann und bestätige dies durch meine zweite Unterschrift. Zur Wahrung der Frist genügt die rechtzeitige Absendung des Widerrufs.

Datum 2. Unterschrift

Gutschein ausfüllen, ausschneiden, in ein Kuvert stecken und absenden an:
Markt&Technik Verlag Aktiengesellschaft, Vertrieb, Postfach 1304, 8013 Haar

HCS18

Fordern Sie mit nebenstehendem Gutschein ein kostenloses Probeheft an. Lernen Sie »Happy-Computer«, das große Heimcomputer-Magazin, völlig unverbindlich kennen.

werk auch das Netzteil zur Versorgung des Laufwerks und des Controllers mit Strom (Bild 3). Für den Betrieb stellt es zwei Spannungen von 12 und 5 Volt bereit, mit denen es die Laufwerkselektronik zur Signalaufbereitung und Motorsteuerung sowie die zwei Motoren speist. Einer der beiden Motoren läßt die in der steifen Plastikhülle der Diskette steckende magnetisierbare Scheibe (Bild 4) mit 300 Umdrehungen pro Minute rotieren. Der zweite Motor, ein sogenannter »Schrittmotor«, dessen Achse sich bei jedem Stromimpuls um einen bestimmten Winkel dreht, ist in der Lage, Bewegungen mit großer Wiederholgenauigkeit auszuführen. Er bewegt einen Schlitten mit dem Schreib-/Lesekopf (mit einem Tonkopf im Kassettenrecorder vergleichbar) in Schritten, die kleiner sind als ein halber Millimeter. Dieser Kopf magnetisiert die metallisch beschichtete Oberfläche der flexiblen Kunststoffolie in bestimmten magnetischen »Mustern«, die beim späteren Lesen wieder einen Sinn ergeben. Die kleinste Bewegung des Kopfschlittens entspricht einer der 40 Spuren (Tracks), auf die wir später noch näher zu sprechen kommen.

Ist nun eine 3-Zoll-Diskette im Laufwerk eingerastet, hat sich zuerst beim Einführen ihre Schutzabdeckung (siehe Bild 1) beiseite geschoben, die Drehachse ist im Mittelloch der Diskette verankert und ein Filz drückt die dünne Plastikfolie von oben auf den unter ihr liegenden Schreib-/Lesekopf. Daraus ergibt sich, daß die Daten immer auf die nach unten gewandte Seite der Diskette geschrieben werden. Ist das Laufwerk in Betrieb, treten zwei Lichtschranken (Bild 3) in Aktion. Die eine tastet den Schreibschutz der Diskette ab, mit der sie sich vor unbeabsichtigtem Beschreiben (und damit auch Löschen) schützen läßt. Die andere erkennt das Indexloch zur Markierung einer bestimmten Stellung der Diskette. Dort nämlich beginnt der erste Sektor jeder Spur. Eine Diskette ist also nicht nur in Spuren unterteilt, sondern zusätzlich in Sektoren (siehe Bild 5). Diese zweite Aufteilung erfolgt jedoch nicht mechanisch wie bei den Spuren, sondern softwaregesteuert. Diese Einteilung der Diskette veranschaulicht der Vergleich mit einer Torte: Stellen Sie sich vor, Sie haben eine Sahnetorte vor sich. Mit einem Messer zerschneiden Sie sie in neun gleich große Stücke. Danach ziehen Sie mit dem Finger 40 von außen nach innen kleiner werdende Kreise. Sie haben jetzt – abgesehen von einem beschmierten Finger – den Aufbau einer Diskette bildlich vor sich. Die Kreise entsprechen den Spuren, die Stücke den Sektoren. Ein solcher Diskettensektor beinhaltet bei normaler Formatierung 512 Byte beziehungsweise ein halbes KByte. Da auf einer Spur jeweils neun Sektoren untergebracht sind, werden den Sektoren zur Unterscheidung beim Formatieren fortlaufende Nummern zugeordnet. Diese Nummern differieren bei den vier Schneider-Formaten. Durch eine Überprüfung dieser Sektornummern (nach dem englischen Begriff »Identifier« auch IDs genannt) erkennt das DOS bei jedem Diskettenzugriff automatisch das momentane Format. »Formatierung« nennt man die Vorbereitung einer noch nicht benutzten, neuen Diskette zum ersten Beschreiben. Bei der Formatierung erfolgt die Einteilung in Spuren und Sektoren. Währenddessen wird die Diskette aber auch auf mechanische Fehler geprüft, die einen späteren Datenverlust zur Folge hätten. Bei der Formatierung ist zwischen diesen vier Formaten zu wählen:

– DATA-Format: Es stellt den größten Speicherplatz aller vier Formate (178 KByte) zur Verfügung. Von einer solchen Diskette läßt sich aber kein CP/M laden (booten). Es besteht aus 40 Spuren zu je neun Sektoren. Diese Sektoren tragen IDs von C1 hex bis C9 hex.

– IBM-Format (genauer gesagt IBM CP/M 86 single sided): Es bietet die geringste Kapazität (152 KByte), hat aber den Vorteil, daß viele andere CP/M-Computer dieses Format (zumindest auf 5¼-Zoll-Disketten) verarbeiten können. Seine geringe Kapazität erklärt sich daraus, daß es auf jeder seiner 40 Spuren nur acht Sektoren mit den Nummern 1 bis 8 bereitstellt.

Bild 4. Hinter der stabilen Plastikhülle verbirgt sich neben aufwendiger Mechanik die Kunststoffolie der Diskette

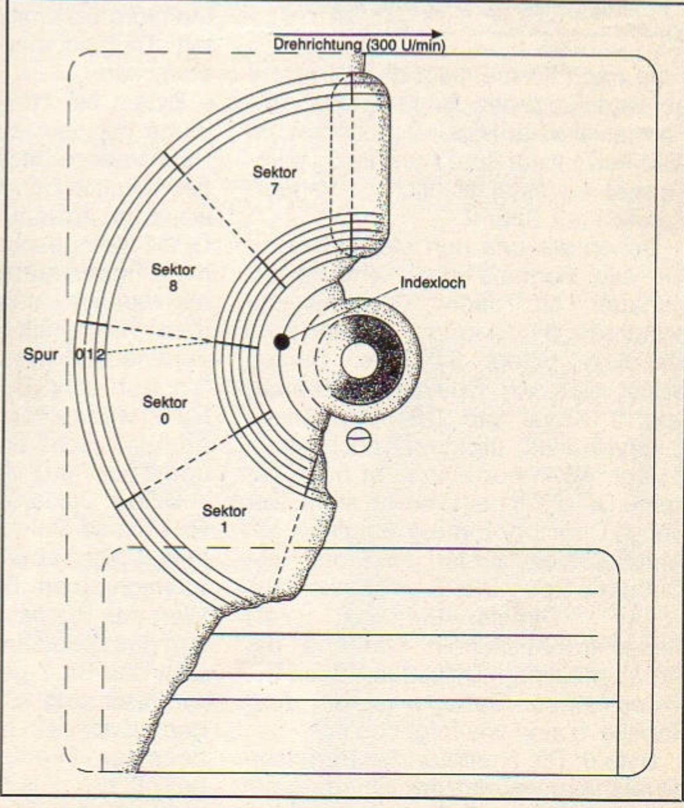


Bild 5. Schematisch betrachtet sieht so die Aufteilung der Diskette nach der Formatierung aus

– CP/M-(System-)Format: Als meistbenutztes Format verfügt es über 169 KByte Speicherplatz. Ähnlich dem DATA-Format besteht auch hier jede Spur aus neun Sektoren. Diese tragen allerdings IDs von 41 bis 49 hex. Die gegenüber dem DATA-Format fehlenden 9 KByte belegt das Betriebssystem CP/M. Da dieses Betriebssystem dem Copyright unterliegt, darf man keine Disketten mit dieser Formatierung weitergeben. Deshalb gibt es als Abwandlung des Systemformats noch das

– Vendor-Format (deutsch: »Verkäufer«-Format): Es ist von der Diskettenaufteilung her mit dem CP/M-Format identisch, jedoch sind die CP/M-Spuren frei. Der Grund dafür sind die genannten rechtlichen Probleme bei der Weitergabe des CP/M.

Jede Diskette besitzt ein eigenes Inhaltsverzeichnis, auch »Directory« genannt. Dieses Directory enthält alle Informationen, die der Computer braucht, um gespeicherte Dateien auf der Diskette zu finden und zu laden. Der Basic-Befehl CAT zeigt auch dem Benutzer die wichtigsten Daten dieses Inhaltsverzeichnisses. Für bessere Übersicht sorgt dabei die wahlfreie Unterteilung des Directory in sogenannte Benutzerbereiche. Der Basic-Befehl IUSER schaltet zwischen diesen 16 Bereichen um.

Wegweiser durch den »Daten-Dschungel«

Je nach Format liegt das Directory an verschiedenen Stellen. Im DATA-Format ist es auf Spur 0 zu finden, im IBM-Format auf Spur 1 und im System-Format (einschließlich Vendor-Format) auf Spur 2.

Sehen wir uns nun stellvertretend für alle Formate das CPM-Format genauer an. Jeder der maximal erlaubten 64 Einträge (Dateien) im Directory belegt 32 Byte. Daraus ergibt sich ein Speicherplatzbedarf von 2 KByte pro Directory. Diese 2 KByte sind übrigens auch dann belegt, wenn nur eine oder noch gar keine Datei auf der Diskette steht. Der erste Directory-Eintrag beginnt am Anfang des ersten Sektors der Directory-Spur. Bild 6 zeigt zwei typische Directory-Einträge als Bildschirm-Ausschnitt während der Arbeit mit einem Disketten-Editor. Die Bytes eines Eintrags (siehe auch Tabelle 1) sind wie folgt codiert:

Byte 0: Die Nummer des Benutzerbereichs, unter der der Eintrag beim Befehl CAT oder DIR angezeigt wird. Ein Wert von E5 hex (wie im zweiten

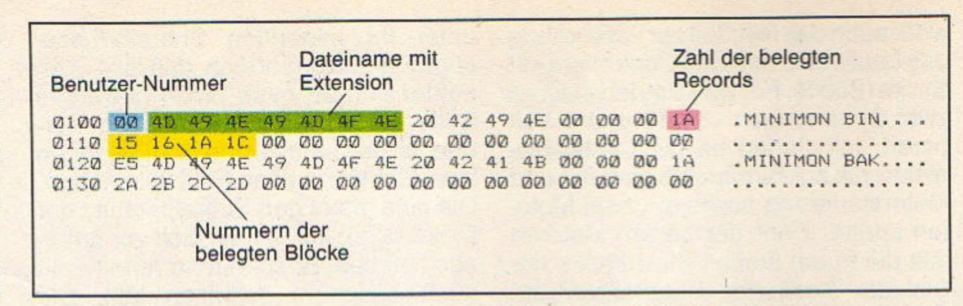


Bild 6. Der Ausschnitt zeigt zwei Einträge einer beispielhaften Directory

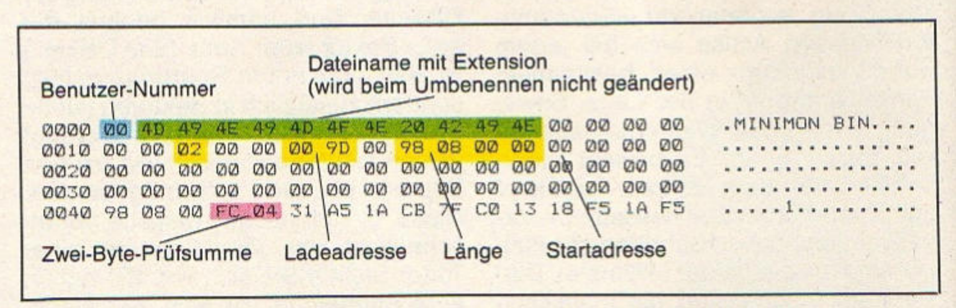


Bild 7. Diese Informationen finden Sie (und das DOS) im Datei-Header

Eintrag in der dritten Zeile von Bild 6) sorgt dafür, daß der Computer die Datei als gelöscht betrachtet und er die durch sie belegten Blöcke der Diskette überschreiben darf. Hat man also eine Diskette mit »IERA,"*.*"« gelöscht, sind keineswegs Daten verloren; die durch Programme belegten Sektoren sind nur im Directory als überschreibbar deklariert. Deshalb ist die versehentliche Eingabe des Löschbefehls auch nicht tragisch, denn durch einfache Änderung dieses Bytes (mit einem Disketten-Monitorprogramm) sind die »gelöschten« Dateien wieder zum »Leben« zu erwecken.

Byte 1 bis 11: Hier steht der Dateiname mit seiner Extension (die drei Buchstaben hinter dem Punkt). Da nur Großbuchstaben vorkommen dürfen, wandeln sowohl Amsdos als auch CP/M kleingeschriebene Namen bei der Eingabe automatisch um. Namen mit weniger als acht Buchstaben vor dem Punkt füllt das DOS ebenfalls automatisch mit Leerzeichen (ASCII 32) auf acht Buchstaben auf. Der Punkt wird nicht mitgespeichert, da er grundsätzlich Verwendung findet und damit nur Platz verschwendet würde.

Byte 9: Deklariert man eine Datei als R/O (Read-Only), ist sie nur noch zu lesen, nicht aber zu löschen oder zu überschreiben. Dazu wird der ASCII-Wert des Buchstaben an dieser Position des Dateinamens um 128 erhöht, also das Bit 7 gesetzt. Zum Beispiel wird aus dem »B« in »TEST.BAS« mit dem Code 42 hex ein »TEST.?AS«, denn das »?« entspricht dem Code C2 hex (42 hex plus 80 hex ergibt nämlich genau C2 hex). Steht eine solche Datei auf der Diskette, ist sie bei CAT durch einen Stern hinter der Extension gekennzeichnet (bei unserem Beispiel »TEST.BAS*«). Eine R/O-Deklaration wird durch ihr Pendant R/W aufgehoben. R/W steht für »Read/Write« und besagt, daß diese Datei jederzeit gelöscht oder überschrieben werden darf. Alle Dateien auf der Diskette befinden sich zunächst im R/W-Status.

Byte 10: Neben der R/O- und R/W-Deklaration gibt es auch den SYS-Status. SYS steht für »System« und besagt, daß der Computer diese Datei zwar als existent ansieht und er sie somit auch laden und starten kann. Bei einem CAT-Befehl jedoch verweigert das Betriebssystem die Anzeige des Eintrags. Diese SYS-Deklaration eignet sich unter anderem dazu, die Übersicht einer Directory zu erhöhen. Besteht beispielsweise ein Programm aus mehreren Teilen wie Basic-Lader, Maschinencode-Laderoutine, Titelbild, Hauptprogramm und eventuell noch einigen Datenfeldern, kann man alle Teile außer dem Basic-Lader als SYS-Dateien kennzeichnen. Bei einem CAT-Befehl erscheint dann statt vier oder mehr Namen nur noch ein einziger auf dem Bildschirm. Das Verfahren der Codierung entspricht dem bei Byte 9 beschriebenen.

Byte 12: In diesem Byte ist festgehalten, um den wievielten Eintrag für die Datei es sich handelt. Ein einzelner Eintrag reicht aus, um Dateien mit einer Länge von bis zu 16 KByte zu verwalten. Für umfangreichere Dateien (ein Bildschirminhalt belegt beispielsweise 17 KByte), benötigt das Betriebssystem mindestens einen weiteren Directory-Eintrag. Damit aber Amsdos die einzelnen Einträge

DELA Elektronik

DRUCKER

Drucker Citizen 120D — **469.00**
incl. Interface

Star NL 10 — **649.00**
incl. Interface und deutschem Handbuch

Star Einzel-Interface	119.00
Einzelblattzuführung	189.00

Dela MP/I/180 — **698.00**
Epson/IBM-kom., 180Zs, 7k-Buffer
NLQ+grafikfähig

Seikosha SL-80AI 24 Nadeldrucker incl. engl. Handbuch	859.00
NEC P6	1149.00
Pin-Freed-Traktor	139.00
Bidirektionaler Traktor	339.00
Einzelblatteinzug	648.00
IBM-Druckerkabel	19.90
Amiga-Druckerkabel	21.90
C 64-Userport-Centronics-Kabel	21.90
Wiesemann-Interface	119.00
Dela-Drucker-Interface	99.00

LAUFWERKE

Amiga 3,5"-Laufwerk — **369.00**

Atari 3,5"-Laufwerk	398.00

DFÜ + BTX

Dataphon S21-23 d — **298.00**

BTX-Term für IBM	198.00
BTX-Term für C64	198.00
Commodore Modem 300 Bd (o. FTZ)	99.00
Universal Modem f. IBM (o. FTZ) 1200 Baud	199.00

DISKETTEN 100er PACK

5"25 MD1D — **69.00**

5"25 MD2D	79.00
5"25 2D HD	390.00
3"5 1DD 135 TPI	269.00
3"5 2DD 135 TPI	279.00
3"0 CF2	650.00

(Bei Ladenverkauf auch 10er Packs zum entsprechenden Preis)

Der Versand erfolgt per Nachnahme (Ausland gegen Vorauskasse + DM 30.00 Versandkosten) + Versandkosten (Selbstkosten). Lieferung freibleibend. Bei großer Nachfrage kann es Verzögerungen geben
Es gelten die gesetzlichen Garantiebestimmungen!

Viele weitere interessante Angebote finden Sie in unseren Filialen.
Besuchen Sie uns doch mal in
Köln 1
Maastrichter Str. 23
Essen
Schützenbahn 11-13
München 22
Bürklein Str. 10

DELA
der Trend-Setter
!!!

Bestellung + Versand
Merkenicher Str. 87-89
5000 Köln 60

ATARI-ZUBEHÖR

Userport für Atari ST — **99.00**

Epromdisk für Atari ST	99.00

simuliert Ramdisk incl. Software (Eprom)
Weiteres Zubehör in unserem Katalog!

AMIGA-ZUBEHÖR

RAM-Erweiterung 512 K — **299.00**
Für Amiga 500

SCHNEIDER

ECB - Adapter CPC 464/646 — **34.50**
Adapter für den Anschluß einer ECB - Karte mit Anschlußpins für ext. +/-12V Versorgung.

ECB - Bus - Platine — **139.00**
Vollgepufferte ECB - Bus - Platine für 7 Steckplätze und zusätzlichen Floppyanschluß. Passend zum Einbau in unser 19" Gehäuse.

19" Gehäuse — **98.00**
Bausatz mit bedruckter Frontplatte
Platz für ECB - Bus + 2 Floppylaufwerke.

Netzteilkarte für 19" Gehäuse — **89.00**
Spannungsreglung für +5V, +12V u. -12V mit Kontrollanzeigen (ohne Trafo)

ECB - Bus Gehäuse — **298.00**
komplett mit Netzteil und Busplatine

Anschlußkabel

-ca. 40 cm für 464, 664	29.50
-für 6128	29.50

PIO - Karte — **89.00**
Digitaler Ein-/Ausgabeport für 3 x 8 Bit TTL-Signale

MONITORE

12" Monochrom-Grün m. Ton — **149.90**

12" TTL Grün		195.00
14" TTL m. Fuß Grün		249.00
dto	Bernstein	249.00
dto	S/W incl. Invers	269.00
NEC Multisync		1449.00
Monitoständer f. 12"		24.90

JOYSTICK/MOUSE

Dela-Micro Fun (6 Microsch.) — **16.90**

IBM Mouse seriell — **119.00**
(incl. Treibersoftware)

Competition Pro	25.90
Quickshot X f. IBM	29.90
Commodore Mouse	79.50

TANDON PC 256 KB:

CPU 8088, IBM PC kompatibel incl. 14" Mono-Monitor, Monochrom Grafikkarte, deutsche Tastatur, MS Dos. 3.1 und GW-Basic mit 2 Floppys a 360 KB — **1869.00**

XPC 10, 10MB Platte, 1 Floppy	2289.00
XPC 20, 20MB Platte, 1 Floppy	2995.00

TANDON PCA 512 KRAM CPU 80286, IBM-AT-kompatibel, 1 Floppy, 1,2 MB incl. 14" Mono-Monitor, Monochrom-Grafikkarte, deutsche Tastatur, MS Dos 3.1+GW Basic

PCA 20 mit 20 MB Platte	4820.00
PCA 30	5555.00

Fordern Sie unseren kostenlosen, farbigen Katalog mit Preisliste schriftlich an

DELA
der Trend-Setter
!!!

BRYLLANTE HANDY-SCANNER

Für IBM — **898.00**

Für Atari	979.00

ZUBEHÖR

CEGA+ -Karte — **339.00**

Public Domain 10 Disk	69.00

Hotline
02 21/7 15 17-0
Bestellung
02 21/7 15 17-20/21/22
Anrufbeantworter
02 21/7 15 17-30
Mailbox (300bd 7/E/1)
02 21/7 15 17-40
Kundenberatung
02 21/7 15 17-50
Telefax
02 21/7 15 17-60

FESTPLATTEN + STEAMER

20 MB Seagate ST 225 — **666.00**
incl. Controller und Kabelsatz

30 MB Seagate ST 238 dto	798.00
Wangtek Seagate Type Streamer 52 MB PAD 5000	1095.00

DELA
unerhört
aktuell und
preiswert

PIO / Relais-Karte — **129.00**
Kombinierte Ein-/Ausgabekarte mit 8 Relais (8 x Umschalter) und 16 Bit TTL-Ein-/Ausgänge. Schaltleistung der Relais ca. 1,5A/220V

RS 232 Schnittstellenkarte — **149.00**
Das Interface zum Betrieb von Modems und Akustikkopplern. Mit Terminalprogrammen für DFÜ - Anwendungen auf Diskette.

unterscheiden kann, sind sie im Byte 12 mit Null beginnend fortlaufend durchnumeriert.

Byte 13 und 14: Sie haben unter Amsdos und CP/M keine Bedeutung.

Byte 15: Hier ist die Zahl der Records (Einheiten zu je 128 Byte) vermerkt, die die Datei belegt. Steht hier eine 80 hex (128 dez), bedeutet dies, daß die Datei noch mindestens einen weiteren Directory-Eintrag besetzt. Die Einheit »Record« besitzt unter Amsdos keine Bedeutung. Sie wurde von den Entwicklern nur wegen des CP/M aus Kompatibilitätsgründen übernommen.

Byte 16 bis 31: Hier stehen die Informationen für das DOS, wo die Datei auf der Diskette zu finden ist. Der Speicherplatz der Diskette ist für jedes der vier Formate in Blöcke zu je 1024 Byte (ein KByte), also zwei Sektoren zu je 512 Byte, aufgeteilt. Daher belegt auch ein noch so kleines Programm auf der Diskette immer mindestens 1 KByte. Der Block entspricht der kleinsten Einheit, die Amsdos verarbeitet. Sie werden jetzt sicher einwenden, daß sich der Speicherplatz der Diskette mit kleineren Blöcken ökonomischer nutzen ließe. Diese Annahme ist sogar richtig – zumindest auf den ersten Blick betrachtet. Bei Verwendung kleinerer Blöcke erhöht sich jedoch zwangsläufig auch deren Zahl. Da die Blöcke dem DOS maßgeblich zur Verwaltung der Diskettendaten dienen, wächst mit steigender Blockzahl natürlich auch der Umfang der Directory immens. Und genau dort würden die eben mühevoll gesparten Byte wieder verbraucht. Sinnvoll ist diese Aufteilung also nur, wenn man mit sehr vielen, kleinsten Dateien arbeitet. Für die normal gebräuchlichen Dateigrößen aber ist die gewählte Blockgröße ein günstiger Kompromiß.

Doch nun wieder zurück zum eigentlichen Thema. Die beiden Directory-Blöcke sind bei allen Formaten einheitlich als Block 0 und 1 codiert. Der erste Block zur Aufnahme von Daten ist also Block 2. Die Nummern der Blöcke, die eine Datei auf der Diskette belegt, stehen in aufsteigender Reihenfolge in den letzten 16 Byte ihres Directory-Eintrags. Belegt sie weniger als 16 Blöcke in einem Eintrag, sind die verbleibenden Byte mit Nullen aufgefüllt.

Der R/O- und SYS-Status läßt sich nicht aus dem Basic heraus erzeugen. Hierzu bedient man sich des transienten CP/M-Befehls STAT oder eines der zahlreichen Hilfsprogramme (Disketten-Monitore beziehungsweise -Editoren), die auch als Public-Domain-Software erhältlich sind.

Im Directory ist keine Unterscheidung zwischen den verschiedenen Dateitypen (Basic-, Binär- oder ASCII-Dateien) getroffen. Diese Informationen stehen im ersten Block der jeweiligen Datei, der den sogenannten Datei-Header enthält. Bild 7 zeigt als Ausschnitt die ersten 80 Byte eines Header-Blocks. Die ersten 66 Byte enthalten alle wichtigen Datei-Informationen. Die einzige Ausnahme machen ASCII-Dateien, für die man keine Informationen wie Dateilänge oder ähnliches benötigt. Sie besitzen also auch keinen Header. Die Byte 0 bis 11 des Datei-Headers sind mit denen des Directory-Eintrags identisch (siehe auch Tabelle 2). Das DOS benutzt sie jedoch nicht. So verändert zum Beispiel ein REN-Befehl (Rename=Umbenennen) nur das Directory, niemals jedoch den Header. Wichtig ist das Byte 18, das die Angabe des Dateityps enthält. Ein Basic-Programm kennzeichnet die Nummer 0. Ist dieses Programm mit »SAVE "NAME",P« gespeichert, also gegen das Listen geschützt, findet sich hier eine 1. Die 2 steht für eine Binärdatei. In den Byte 21 und 22 ist (in der Reihenfolge High-, Low-Byte) die Ladeadresse angegeben. Für Basic-Programme beispielsweise steht hier 170 hex, gespeicherte Bildschirminhalte beginnen bei C000 hex. Byte 24 und 25 geben die Länge eines Programms an. Diese ist natürlich ebenso variabel (abhängig vom Programm) wie die Ladeadresse. In Byte 26 und 27 folgt nun bei Maschinencode-Programmen die Startadresse, sofern sie beim Speichern angegeben wurde. Diese Angabe ist nötig, will man ein solches Programm mit »RUN "NAME"« starten. Alle anderen Byte des Datei-Headers sind vom Betriebssystem nicht belegt und deshalb mit Null-Bytes aufgefüllt. Die folgenden 2 Byte sind eine Prüfsumme, die sich aus der Addition der vorangegangenen 66 Byte berechnen. Anhand dieser Prüfsumme unterscheidet der Computer auch zwischen ASCII-Dateien und anderen. Stimmt sie nämlich nicht, liegt eine Text-Datei vor. Beim Laden eines Programms mit Header werden die beschriebenen Daten mit in den Arbeitsspeicher (RAM) des CPC geladen. Die Ladeadresse für die einzelnen Header-Daten variiert. Die wichtigsten Angaben jedoch, Dateityp und Adressen beispielsweise, stehen ab AE42 hex. Sie lassen sich also sehr einfach für eigene Zwecke verwenden.

In erster Linie wird eine Diskette zum Speichern von Programmen mit dem Basic-Befehl »SAVE "NAME"« genutzt. Sie können ein Basic-Programm aber auch als ASCII-Datei speichern, indem Sie den Befehl »SAVE "NAME",A« (»A« für »ASCII«) verwenden. Auch haben Sie die Wahl, ein Basic-Programm so zu speichern, daß man es nach dem Laden nicht mehr listen kann. Die Befehlsfolge hierfür lautet »SAVE "NAME",P«. (Das »P« steht für »Protected«, also deutsch: geschützt). Bei praktischen Anwendungen (Textverarbeitung, Datenverwaltung und ähnlichem) werden aber oft weniger Programme als vielmehr Daten auf Diskette ausgelagert. Will man zum Beispiel die Inhalte der beiden Textvariablen <a$> und <b$> auf Diskette ablegen, benutzt man folgende Programmzeilen:

```
OPENOUT "NAME"
PRINT #9,a$,b$
CLOSEOUT
```

Disketten in der Praxis

Für das Laden der so gespeicherten Daten benutzt man diese Zeilen:

```
OPENIN "NAME"
INPUT#9,a$,b$
CLOSEIN
```

Mit der beschriebenen Methode lassen sich natürlich auch numerische Variablen (die Zahlenwerte beinhalten) auf Diskette speichern. Um einen Text einzulesen, dessen Länge nicht bekannt ist, findet folgendes Programm Verwendung:

```
OPENIN "NAME"
    WHILE NOT EOF
        INPUT #9,a$
        PRINT a$
    WEND
 CLOSEIN
```

Die Systemvariable EOF (»End Of File«, also »Ende der Datei«) nimmt den Wert -1 an, wenn der letzte Datensatz der Datei gelesen ist, ansonsten ist sie gleich Null.

Will man einen Teil des Arbeitsspeichers auf Diskette ablegen, benutzt man den Befehl

```
SAVE "NAME",b,anfang,länge,start
```

wobei <anfang> der Adresse des ersten Byte entspricht und <länge> der Anzahl der Bytes. <start> gibt die Startadresse an, bei der ein Maschinencode-Programm nach dem Laden startet. Oft zum Einsatz kommt das Speichern von Computergrafiken. Die Befehlsfolge

```
SAVE "BILDNAME",b,&C000,&4000
```

legt das sichtbare Monitorbild unter dem Dateinamen »BILDNAME« auf Diskette ab. Zum Laden genügt ein einfaches

```
LOAD "BILDNAME"
```

Es ist wichtig darauf zu achten, daß sich vor dem Laden eines Bildes der

Bildschirm nicht nach oben verschoben hat, also gescrollt wurde. Daher müssen Sie vor dem Laden des Bilds mit einem MODE-Befehl den Modus setzen, in dem es gespeichert wurde. Haben Sie vor dem Speichern die Farbregister des Computers mit INK-Befehlen verändert, müssen Sie diese auch vor dem Laden wieder eingeben, da sie nicht mit gespeichert sind.

Um auf einen anderen Benutzer-Bereich der Diskette zu wechseln (siehe oben), verwenden Sie den Befehl

```
|USER,nummer
```

Der erlaubte Wertebereich der Variablen <nummer> liegt zwischen 0 und 15.

Wollen Sie eine Datei auf der Diskette umbenennen, benutzen Sie den Befehl REN (steht für Rename). Er verlangt als Parameterübergaben den neuen und den alten Dateinamen.

```
a$="altname"
b$="neuname"
|REN,@b$,@a$
```

Auch das gezielte Löschen einer Datei von der Diskette ist kein Problem:

```
a$="BEISPIEL.EXT"
|ERA,@a$
```

löscht nur die Datei mit dem Namen »BEISPIEL.EXT«. Wahlweise lassen sich aber auch mehrere Dateien mit einem Befehl löschen. Um beispielsweise alle Backup-Dateien (.BAK) zu löschen, die bei mehrmaligem Speichern eines Programms unter demselben Namen automatisch entstehen, genügt die Anweisung

```
a$="*.BAK"
|ERA,@a$
```

Das Zeichen »*« ist eine sogenannte »Wildcard« oder auch »Joker«. Es steht für eine beliebige Anzahl und Art von Buchstaben. Also löscht das Betriebssystem mit diesem Befehl alle Dateien, deren Extension »BAK« lautet. Eine komplette Diskette (nämlich alle Dateien) löscht demzufolge

```
a$="*.*"
|ERA,@a$
```

Die zweite Wildcard, »?«, steht stellvertretend für einen einzelnen, beliebigen Buchstaben. So spricht »*.BA?« alle Dateien an, deren Extension mit der Buchstabenkombination »BA« beginnt; also nicht nur die Backups (»BAK«) sondern zum Beispiel auch Basic-Programme mit der Extension »BAS«.

Weitgehend unbekannt ist die Fähigkeit des Amsdos, wahlweise ein Directory selektierter Dateien auszugeben. Anstelle des CAT-Befehls verwendet man dann einfach den Befehl |DIR. Allerdings fehlt gegenüber CAT die Angabe der Dateilängen auf dem Bildschirm. Wollen Sie sich einen Überblick über alle Basic-Programme auf einer Diskette verschaffen, verwenden Sie die Befehle

```
a$="*.BAS"
|DIR,@a$
```

Byte	Beschreibung
0	Benutzer-Nummer (E5 hex bedeutet gelöscht)
1 bis 11	Name mit Extension (ohne Punkt)
12	Nummer des Directory-Eintrags (beginnt bei 0)
13 bis 14	unbenutzt (immer 0)
15	Anzahl der belegten Records (zu je 128 Byte)
16 bis 31	belegte Blöcke, mit Nullen aufgefüllt

Tabelle 1. Byte-Belegung eines Directory-Eintrags

Byte	Beschreibung
0	Benutzer-Nummer
1 bis 11	Name mit Extension (ohne Punkt)
18	Dateityp 0=Basic-Programm 1=geschütztes Basic-Programm 2=Binär
21 und 22	Ladeadresse des Programms (170 hex für Basic)
24 und 25	Dateilänge
26 und 27	Startadresse des Programms
67 und 68	Prüfsumme über die ersten 66 Byte des Headers

Tabelle 2. Byte-Belegung eines Datei-Headers

Besitzer zweier Diskettenlaufwerke schalten mit »|A« und »|B« zwischen den beiden Laufwerken hin und her. Aber auch mit gewähltem A-Laufwerk läßt sich ohne Umschaltung direkt vom Laufwerk B laden:

```
LOAD "B:NAME"
```

Im Dateinamen darf man bei »LOAD« auch noch die Nummer des Benutzer-Bereichs angeben, in dem eine Datei gespeichert ist. Soll beispielsweise ein Programm aus dem Benutzer-Bereich 12 vom B-Laufwerk geladen werden, lautet der Befehl

```
LOAD "12B:NAME"
```

Ebenfalls vielfach unbekannt ist der Befehl DRIVE. Mit ihm läßt sich, abhängig vom Inhalt einer Variablen, das Laufwerk selektieren.

```
a$="B"
|DRIVE,@a$
```

bewirkt dasselbe wie »|B«.

Bei den CPCs 664 und 6128 darf man den Amsdos-Befehlen Dateinamen auch direkt übergeben. Also lassen sich Befehlsfolgen wie

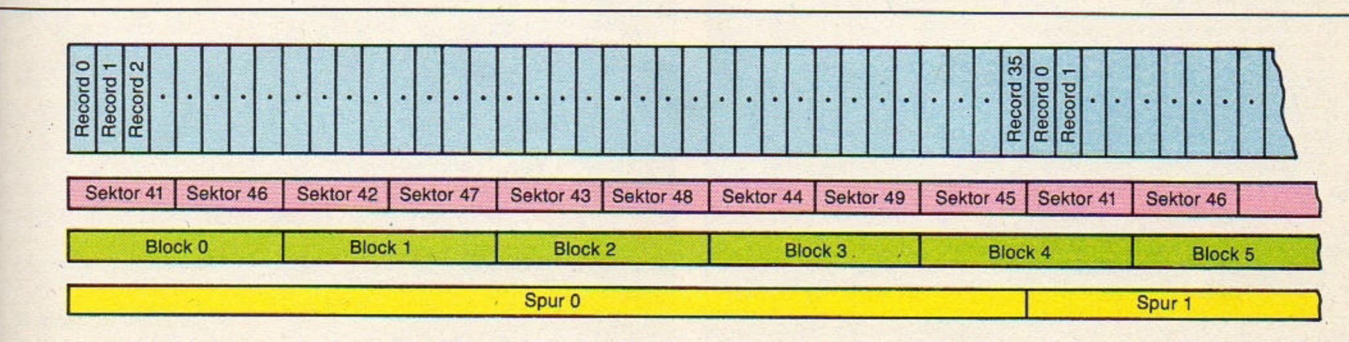
```
a$="*.BAK"
|ERA,@a$
```

durch

```
ERA,"*.BAK"
```

ersetzen. Wenn Sie noch tiefer in den Umgang mit den Schneider-Diskettenlaufwerken einsteigen wollen, folgen Sie uns bitte im zweiten Teil dieses Beitrags ab Seite 38 auf unserer Exkursion in die vielfältigen Fähigkeiten des DOS und Controllers.

(Oliver Suttorp/ja)



Eine Diskette im CPC-Systemformat besteht inklusive der Directory und beider Systemspuren aus:
1440 Records zu je 128 Byte
360 Sektoren zu je 512 Byte (vier Records)
180 Blöcken zu je 1024 Byte (acht Records beziehungsweise zwei Sektoren)
40 Spuren zu je 4608 Byte (4,5 Blöcke beziehungsweise neun Sektoren beziehungsweise 36 Records)

Bild 8. Schematische Unterteilung einer Spur des CP/M-Formats

Was die Floppy sonst noch alles kann

Wenn Sie sich bereits den ersten Teil des Beitrags einverleibt haben, verfügen Sie über die nötigen Grundlagen für diese Fortsetzung. Wir steigen nun gemeinsam hinab in die Tiefen des DOS.

Die in der Einführung ab Seite 31 dieser Ausgabe beschriebene Art der Diskettenprogrammierung ist die einfachste und oberflächlichste. Wir wollen sie als Stufe 1 bezeichnen, denn hier nimmt das Betriebssystem dem Anwender den Großteil der Arbeit aus den Händen. Er muß sich nur darum kümmern, genügend Platz auf einer formatierten Diskette bereitzustellen. Von den Problemen, die das DOS automatisch bewältigt, bemerkt er nichts. Ähnlich sieht es in der Stufe 2 aus. Sie ist nicht mehr in Basic zu programmieren, man muß sich auf die Ebene der Maschinensprache-Programmierung hinunterbegeben. Aber keine Angst – durch die im Betriebssystem enthaltenen Vektoren gestaltet sich diese Programmierung recht einfach. Auch hier brauchen Sie sich nicht um die Verwaltung der Diskette zu kümmern. Sie müssen lediglich der jeweiligen ROM-Routine die nötigen Parameter übergeben, den Rest besorgt das DOS.

Tabelle 1 zeigt eine Liste der wichtigsten Vektoren mit ihren Übergabeparametern. Damit läßt sich wie mit den äquivalenten Basic-Befehlen arbeiten, wenn man über zumindest rudimentäre Kenntnisse der Assembler-Programmierung verfügt. Als Beispiele mögen Ihnen Listing 1 und 2 dienen, die jeweils eine (Binär-) Datei laden beziehungsweise speichern. Wie Sie sehen, lassen sich mit dieser Art der Programmierung Speicherinhalte fast so einfach wie unter Basic speichern und laden. Sie dürfen die Dateien jedoch im Gegensatz zum Basic an jede beliebige Speicheradresse laden. Sie müssen dabei nur darauf achten, keine Betriebssystem-Bereiche und -Variablen zu überschreiben. Der Speicher von Adresse 40 hex bis etwa A600 hex und von C000 hex bis FFFF hex steht normalerweise frei zur Verfügung.

Mit der Vektorenbenutzung ist es noch immer nicht möglich, auf Daten einer Diskette direkt zuzugreifen. Immer noch kümmert sich das Amsdos darum, auf welche Sektoren die Daten »wandern«. Das ändert sich erst in der Stufe 3 der Diskettenprogrammierung. Sie geht auf alle Wünsche des Anwenders hinsichtlich des Diskettenzugriffs ein, erfordert aber fortgeschrittene Kenntnisse der Assembler-Programmierung sowie tiefgreifendes Wissen um die Datenorganisation. Den letzteren Aspekt wollen wir Ihnen vermitteln. Im folgenden erfahren Sie deshalb, wie Sie auf einzelne Sektoren und Spuren zugreifen, Spuren formatieren und Fremdformate verarbeiten. Nehmen Sie keine Diskette mit wichtigen Daten als Versuchsobjekt, denn es ist zu mindestens 95 Prozent sicher, daß solche Daten schon bei den ersten Versuchen das Zeitliche segnen.

```
; Laden einer Binärdatei in Assembler
; entspricht dem Basic-Befehl ))LOAD "TEST.BIN"((
;
LOAD    LD      HL,NAME  ; Startadresse des Namens
        LD      B,8      ; Anzahl der Buchstaben
(       LD      DE,&4000 ; Puffer, in diesem Fall nicht nötig)
        CALL    &BC77    ; Eröffne Eingabedatei
;
        EX      DE,HL    ; Original-Ladeadresse übernehmen
(       LD      HL,&C000 ; Beispiel: Ladeadresse C000 hex)
        CALL    &BC83    ; gesamte Datei laden
;
        CALL    &BC7A    ; Eingabedatei schließen
;
NAME    DEFM    'TEST.BIN'
```

Listing 1. Eine Assembler-Routine zum Laden einer Binärdatei via Systemvektoren

Adresse (hex)	Parameter	Beschreibung
BC77	E:B = Länge des Dateinamens DE = Adresse eines 2-KByte-Puffers HL = Adresse des Dateinamens A:NC = Fehler aufgetreten HL = Adresse des Datei-Headers DE = Original-Ladeadresse der Datei	Eröffnen einer Datei zum Lesen
BC83	E:HL = Ladeadresse A:NC = Fehler aufgetreten	Einlesen einer geöffneten Datei (kein Puffer nötig)
BC80	E: – A:A = Zeichen (1A hex=EOF) NC = Fehler aufgetreten	Lesen eines Zeichens aus einer (ASCII-)Datei (Puffer nötig)
BC7A	E: – A:NC = Fehler aufgetreten	Schließen einer Eingabedatei
BC8C	E:B = Länge des Dateinamens DE = 2-KByte-Puffer HL = Adresse des Dateinamens A:NC = Fehler aufgetreten HL = Adresse des Datei-Headers	Eröffnen einer Ausgabedatei

Adresse (hex)	Parameter	Beschreibung
BC98	E:A = 0 für Basic-Programm = 1 für geschütztes Basic-Programm = 2 für Binärdatei = 6 für ASCII-Datei HL = Anfangsadresse DE = Anzahl der Byte BC = Startadresse (nur bei Maschinencode-Programmen) A:NC = Fehler aufgetreten	Schreiben eines kompletten Speicherbereichs (kein Puffer nötig)
BC95	E:A = Zeichen A:NC = Fehler aufgetreten	Schreiben eines Zeichens in den Ausgabepuffer
BC8F	E: – A:NC = Fehler aufgetreten	Schließen einer Ausgabedatei
BC9B	E:DE = 2-KByte-Puffer	CATalog
E: und A: stehen für Übergabeparameter (Ein- und Ausgabe); NC bedeutet »No Carry-Flag« A, BC, DE und HL sind die Z80-Register		

Tabelle 1. Diese Firmwarevektoren dienen der Disketten-Dateiverwaltung

Professionelle Grafikprogramme für Schneider CPC 6128 + Joyce

Markt&Technik
CPC/Joyce-Software
DIGITAL RESEARCH
DR Draw™
FROM DIGITAL RESEARCH THE CREATORS OF CP/M
DR Draw: macht aus Ihren Ideen ein Kunstwerk.
3"-Diskette für den Schneider CPC 6128 und Joyce PCW 8256

DR Draw: Macht aus Ihren Ideen ein Kunstwerk.

Verwenden Sie DR Draw, um Organisations-Diagramme, Flußdiagramme, Logos, technische Zeichnungen, Schaubilder, Platinenentwürfe und jede nur erdenkliche Art von Linien- und Formgrafiken zu entwerfen. Jeder Bestandteil Ihrer Zeichnung kann auf vielfältige Weise durch Farben und Schraffuren hervorgehoben werden.

Die Fähigkeiten auf einen Blick:

- Erstellung beliebiger Zeichnungen
- vorprogrammierte Figuren wie Kreise, Quader, Rechtecke, Kreisbögen, Polygone und Linien
- freie Wahl der Gestaltungselemente wie Farben, Muster und Schriftarten
- Vergrößerungen und Ausschnittdarstellungen
- Teile einer Zeichnung können kopiert, verschoben oder gelöscht werden
- Grafiken können gespeichert, geplottet oder gedruckt werden
- einfache Bedienung durch Menüauswahl

Hardwarevoraussetzungen:

DR Draw läuft auf jedem Schneider CPC 6128 oder Joyce PCW 8256 mit einem oder zwei Diskettenlaufwerken. Die Grafiken können auf jedem Drucker oder Plotter ausgegeben werden, für den ein GSX-Treiber verfügbar ist. Dazu zählen Schneider-, Epson- und Shinwa-Drucker sowie der Plotter HP 7470A.

Diese Markt & Technik-Software-Produkte erhalten Sie in den Computer-Abteilungen der Warenhäuser, bei Ihrem Computerfachhändler, im Buchhandel oder direkt beim Verlag gegen Vorauskasse.

DR Graph: Präsentationsgrafiken mit professionellem Niveau.

DR Graph ist ein interaktives Softwarepaket, mit dem Sie Ihren Mikrocomputer zur Erstellung von Geschäftsgrafiken und Text-Charts verwenden können.

Die Fähigkeiten auf einen Blick:

- Linien-Grafiken, Histogramme, Torten-Grafiken, Stufen-Grafiken, Strich-Histogramme, Punkte-Grafiken und Text-Grafiken
- freie Wahl der Gestaltungselemente wie Beschriftungen, Titelzeilen, Legenden, Farben, Schriftarten und Ränder
- frei wählbare Skalierung
- variable Linien- und Balkenbreite
- Schnittstelle zu anderen Programmen
- beliebig positionierbare Anmerkungen
- Grafiken können gespeichert, geplottet oder gedruckt werden
- einfache Bedienung durch Menüauswahl

Hardwarevoraussetzungen:

DR Graph läuft auf jedem Schneider CPC 6128 oder Joyce PCW 8256 mit einem oder zwei Diskettenlaufwerken. Die Grafiken können auf jedem Drucker oder Plotter ausgegeben werden, für den ein GSX-Treiber verfügbar ist. Dazu zählen Schneider-, Epson- und Shinwa-Drucker sowie der Plotter HP 7470A.

	Version	Best.-Nr.	Format	Preis DM	sFr	öS
DR Draw	CPC 6128/Joyce	51613	3"	199,-*	178,-	1990,-*
DR Graph	CPC 6128/Joyce	51614	3"	199,-*	178,-	1990,-*

* inkl. MwSt. Unverbindliche Preisempfehlung

Markt&Technik Verlag AG, Buchverlag, Hans-Pinsel-Straße 2, 8013 Haar bei München, Telefon (089) 4613-0

Bestellungen im Ausland bitte an: SCHWEIZ: Markt&Technik Vertriebs AG, Kollerstrasse 3, CH-6300 Zug, Telefon (042) 415656 · ÖSTERREICH: Rudolf Lechner & Sohn, Heizwerkstraße 10, A-1232 Wien, Telefon (0222) 677526 · Ueberreuter Media Verlagsges. mbH (Großhandel), Alser Straße 24, A-1091 Wien, Telefon (0222) 481538-0.

```
; Speichern einer Binärdatei in Assembler
; entspricht dem Basic-Befehl ))SAVE "TEST.BIN",B,
  &C000,&3FFF,&C000((
;
SAVE    LD      HL,NAME  ; Adresse des Namens
        LD      B,8      ; Anzahl der Buchstaben
(       LD      DE,&4000 ; Puffer, in diesem Fall nicht nötig)
        CALL    &BC8C    ; Eröffne Ausgabedatei
;
        LD      A,2      ; Kennzeichen für Binärdatei
        LD      HL,&C000 ; ab Adresse C000 hex
        LD      DE,&3FFF ; 3FFF hex Byte speichern
(       LD      BC,&C000 ; Startadresse, wenn Programm)
        CALL    &BC98    ; gesamte Datei schreiben
;
        CALL    &BC8F    ; Schließe Ausgabedatei
;
NAME    DEFM    'TEST.BIN'
```

Listing 2. Das Programm zum Speichern unterscheidet sich nur wenig von der Laderoutine

Wie Sie bereits wissen, ist eine Diskette in Spuren und Sektoren unterteilt. Normalerweise besteht jede Spur aus neun Sektoren zu je 512 Byte. Für die Unterscheidung dieser neun Sektoren ordnet der Computer jedem Sektor einer Spur beim Formatieren eine eigene Nummer zu. Diese sogenannte ID steht direkt vor dem eigentlichen Datenfeld. Das vier Byte große ID-Feld hilft dem Controller bei der Identifizierung des folgenden Sektors.

Befehl	Adresse	Parameter	Beschreibung
81 hex	CA72 hex	E:A=0 (Meldungen aus) A=FF hex (Meldungen ein)	Ausschalten der Disketten-Fehlermeldungen
82 hex	C60D hex	E:HL=Tabellenadresse	Laufwerks-Parameter ändern
		Byte der Tabelle	**Zweck**
		1 und 2 3 4 5 6 7 8	Motor-Start-Zeit (1/50 Sek.) Motor-Stop-Zeit (1/50 Sek.) GAP-Länge (normal AF hex) nicht benutzt Steprate in ms (normal 12) Head-Unload-Time (32 ms) Head-Load-Time (16 ms)
83 hex	C581 hex	E:A=Format (ID) 1: IBM 65: System 93: DATA	Einstellen der Format-Parameter im RAM
84 hex	C666 hex	E:E=Drive D=Spur C=Sektor HL=Puffer	Lesen eines Sektors von Diskette
85 hex	C64E hex	E:E=Drive D=Spur C=Sektor HL=Puffer	Schreiben eines Sektors
86 hex	C652 hex	E:E=Drive D=Spur C=erster Sektor der Spur HL=Adresse der Parametertabelle:	Spur formatieren
		Byte	**Bedeutung**
		1 2 3 4 5 und so weiter für alle Sektoren	Spur Kopfnummer (0) Sektor-ID Sektorgröße (2) Spur
87 hex	C763 hex	E:D=Spur	positioniere Kopf über Spur
88 hex	C630 hex	E:A=Laufwerk (0 für A; 1 für B) A:C=Laufwerk ready	Laufwerk verfügbar?
89 hex	C603 hex	E:A=Anzahl (0=256)	Anzahl der Leseversuche bei Lesefehlern

E: und A: stehen für Übergabeparameter (Ein- und Ausgabe)
C bedeutet gesetztes Carryflag
A, BC, DE, und HL sind die Z80-Register

Tabelle 2. Die »verborgenen« RSX-Befehle des Amsdos sind nur per Maschinen-Programmierung zugänglich

Im ersten Byte ist zur Kontrolle noch einmal die momentane Spurnummer vermerkt. Das zweite Byte ist für den Betrieb der Schneider-Laufwerke uninteressant, da sie nur mit einem Schreib-/Lese-Kopf arbeiten. Deshalb steht hier immer eine Null. Bei Doppelkopf-Laufwerken bestimmt dieses Byte, auf welche Diskettenseite zugegriffen wird. Das dritte Byte ist nun die Sektor-ID, die man beim Lesen und Schreiben eines Sektors natürlich immer mit angeben muß. Das DOS vergleicht die gesuchte ID mit der des aktuellen Sektors. Sind die Werte identisch, greift es auf diesen Sektor zu, ansonsten sucht es weiter. Das vierte und letzte Byte ist das interessanteste von allen. Mit ihm läßt sich die physikalische Größe eines Sektors festlegen. Man ist nämlich nicht auf 512 Byte pro Sektor beschränkt, sondern kann zwischen 256, 512, 1024, 2048 und 4096 Byte wählen. Da für diese Kapazitäts-Festlegung nur ein Byte zur Verfügung steht, erfolgt sie verschlüsselt. Den genannten Werten entsprechen in dieser Reihenfolge die Zahlen 1, 2, 3, 4 und 5.

Um dem Programmierer auch in dieser Stufe die Arbeit zu erleichtern, haben sich die Programmierer des Amsdos etwas Besonderes einfallen lassen: Jede wichtige Funktion ist als RSX-Befehl ins Betriebssytem eingebunden. Diese Befehle in Form einer »residenten System-Erweiterung« erlauben einen komfortablen Aufruf der Routinen. Doch noch einen weiteren wichtigen Vorteil bietet diese Technik. Die Adresse des Befehls läßt sich nämlich durch eine Betriebssystem-Routine (KL-FIND-COMMAND) automatisch berechnen. Falls es jemals eine geänderte Version des Amsdos

```
10 ' ASCII-Disk-Monitor                          [E2FC]
20 '                                             [6052]
30 ' Oliver Suttorp, Johannisstr. 64             [00E6]
40 ' 5024 Pulheim, Tel.: 02238/56368             [817A]
50 '                                             [6158]
60 MEMORY &8FFF                                  [B15E]
70 trackadr=&A010:sekadr=&A012                   [11E6]
80 FOR i=&A000 TO &A01A:READ a$                  [8B26]
90 POKE i,VAL("&"+a$):NEXT                       [E068]
100 DATA 21,1A,A0,CD,D4,BC,22,1B,A0              [68BA]
110 DATA 79,32,1D,A0,1E,00,16,00,0E              [6132]
120 DATA 00,21,00,90,DF,1B,A0,C9,84              [C55E]
130 MODE 2:INK 0,0:INK 1,26:BORDER 0             [EC78]
140 OPENOUT "d":CLOSEOUT:' Login                 [3C92]
150 LOCATE 1,23:INPUT "Track :(0-39) ";t
    rack                                         [D2AE]
160 LOCATE 1,24:PRINT"Sektor :";                 [A9D4]
170 IF PEEK(&A895)=179 THEN PRINT"(&c1-&
    c9)";                                        [CB7A]
180 IF PEEK(&A895)=170 THEN PRINT"(&41-&
    49)";                                        [0EAE]
190 IF PEEK(&A895)=155 THEN PRINT"(1-8)"
    ;                                            [124C]
200 INPUT sek                                    [B04A]
210 POKE trackadr,track:POKE sekadr,sek          [1302]
220 CALL &A000                                   [7BCE]
230 LOCATE 1,1:FOR i=&9000 TO &91FF              [9FE0]
240 a=PEEK(i):IF a<32 THEN a=46                  [802A]
250 PRINT CHR$(a);:NEXT                          [6696]
260 GOTO 150                                     [C24E]
```

Listing 3. Dieser ASCII-Monitor zeigt Ihnen den Inhalt eines beliebigen Diskettensektors

```
; Lesen eines Diskettensektors in Maschinensprache
;
        LD      HL,BEFEHL   ; Adresse des Befehlscode
        CALL    &BCD4       ; Suche RSX-Befehl (KL-FIND-COMMAND)
        LD      (ADR),HL    ; Adresse speichern
        LD      A,C
        LD      (ROMNR),A   ; ROM-Nummer sichern (normal 7)
;
        LD      E,0         ; Drive nach E (0=A; 1=B)
        LD      D,1         ; Spurnummer nach D (0 bis 42)
        LD      C,1         ; Sektor-ID (-Nummer)
        LD      HL,BUFFER   ; Adresse des Puffers nach HL
        RST     &18         ; FAR-CALL zur gesuchten Adresse
DEFW    ADR                 ; Zeiger auf Adresse
        RET                 ; und zurück
BUFFER  DEFS    &200        ; 512-Byte-Puffer für Sektorinhalt
ROMNR   DEFS    1           ; ein Byte Speicher für ROM-Nummer
BEFEHL  DEFB    &84         ; RSX-Befehlscode (Sektor lesen)
```

Listing 3a. Der Assembler-Quellcode der kurzen Maschinencode-Routine im ASCII-Monitor zeigt, wie man in Maschinensprache einzelne Sektoren liest

geben sollte, sind dann auch noch »alte« Programme ohne Anpassung lauffähig.

Die insgesamt neun RSX-Befehle stehen in Tabelle 2. Bei der Durchsicht dieser Tabelle werden Sie feststellen, daß es jetzt bei den Diskettenzugriffen nicht mehr um Dateien geht. Auch Directory oder Blöcke sind für diese Routinen ein Fremdwort. Vielmehr richten sich Zugriffe jetzt gezielt auf einzelne Sektoren der Diskette, was Sie anhand einiger kleiner Beispiele nachvollziehen können. Listing 3 liest einen Sektor von Diskette und stellt dessen Inhalt in Form von ASCII-Zeichen auf dem Bildschirm dar. Es besteht aus dem Basic-Hauptprogramm und einer wenige Byte umfassenden Maschinencode-Routine zum Lesen des Sektors. Den ausführlich dokumentierten Assembler-Quellcode dieser Routine zeigt Listing 3a. Die Ermittlung der Befehlsadresse mag Ihnen zunächst etwas aufwendig erscheinen, sie hat aber wie bereits weiter oben erwähnt den Vorteil, daß das Programm auf allen ROM-Versionen läuft, also beispielsweise auch unter dem VDOS der Vortex-Laufwerke.

Der Rest der Routine besteht im wesentlichen nur noch aus dem Laden der benötigten Register zur Parameterübergabe und dem Aufruf der Betriebssystemroutine mittels eines FAR-CALL (RST 18). Das auf den RST-Befehl folgende Byte deutet auf eine Tabelle, in der die Adresse und die ROM-Nummer der Routine stehen. In diesem Fall lautet die Adresse C666 hex im ROM Nummer 7. Diese »umständliche« Programmierung hat den weiteren Vorteil, daß durch einfache Änderung eines einzelnen Byte, des Befehls-Byte (letztes Byte in den DATA-Zeilen), in 85 hex der Sektor nicht gelesen, sondern geschrieben wird. Der Aufbau des Basic-Programms ist sehr leicht überschaubar. Zeile 140 dient dazu, das RAM mit den Formatwerten der derzeitigen Diskette zu laden. Die Zeilen 170 bis 190 stellen anhand der Zahl der freien Blöcke fest, welche IDs auf der Diskette Verwendung finden. Die Angabe falscher IDs führt nämlich zur Fehlermeldung »READ FAIL«, da dann der FDC (»Floppy-Disc-Controller«) vergeblich nach dem Sektor sucht.

Damit sind wir bei den RAM-Speicherbereichen des DOS angekommen, denen wir einen wesentlichen Teil unserer Aufmerksamkeit schenken müssen. Sie enthalten Informationen, die sonst nur schwer zugänglich sind. So halten wir den Programmieraufwand in Grenzen. Wurde einmal unter Basic auf die eingelegte Diskette zugegriffen, ist sie »eingelogt«, das heißt die Formatparameter des augenblicklichen Diskettenformats sind im RAM abgelegt. Das »Login« einer Diskette geschieht am einfachsten mit der Befehlsfolge

```
OPENOUT "DUMMY"
CLOSEOUT
```

(Die Bezeichnung »DUMMY« steht für einen beliebigen Dateinamen und besagt, daß diese Datei nur temporär genutzt wird und deshalb hinterher auch nicht im Directory steht.) Jetzt sind im DPB (Disk Parameter Block) wichtige Informationen zu finden, die sich ohne Probleme unter Basic mit PEEK auswerten lassen. In Adresse A895 hex steht beispielsweise die maximale Anzahl der Blöcke der eingelegten Diskette. Über diesen Umweg kann man das Format der Diskette feststellen, da jedes der Diskettenformate eine andere Kapazität hat (170 entspricht dem System- und Vendor-, 179 dem DATA- und 155 dem IBM-Format). Ab Adresse A8B9 hex liegt die sogenannte »Allocation Table«, in der 22 Byte die Belegung der Diskette codieren. Gesetzte Bits stehen für belegte Blöcke, nicht gesetzte für freie. Mit dem Umweg über diese Tabelle ist der freie Speicherplatz einer Diskette aus dem Basic leicht festzustellen. Das ist wichtig, will man aus einem Programm heraus Daten auf Diskette speichern. Ist die Diskette nämlich voll, führt der Versuch zur DOS-Meldung »Disk full« und zur Unterbrechung des laufenden Programms. Eine Liste der wichtigsten DPB-Adressen ist in Tabelle 3 zu finden. Ein Beispiel zur automatischen Formaterkennung und Ermittlung des freien Speicherplatzes sehen Sie im Listing 4. Dieses Programm macht ausschließlich Gebrauch von den beschriebenen Adressen.

Mit dem ASCII-Monitor aus Listing 1 lassen sich, wie Sie wissen, bestimmte Sektoren der Diskette lesen beziehungsweise schreiben. Doch

Adresse (hex)	Normalbelegung	Bedeutung
A890 bis A891	36	Records pro Spur (9 Sektoren mal 4 Records)
A892	3	Block-Shift
A893	7	Block-Maske
A894	0	Extent-Maske
A895 bis A896	170	maximale Zahl der Blöcke:
		170=System und Vendor 179=DATA 155=IBM
A897 bis A898	63	maximale Zahl der Directory-Einträge minus 1
A899 bis A89A	C000 hex	Directory-Größe (codiert: entspricht 2 Blöcken, also 2 KByte)
A89B bis A89C	16	Directory-Einträge pro Record
A89D bis A89E	2	vom System belegte Spuren: 2=System und Vendor 0=DATA 1=IBM
A8B9 bis A8CE	-	Allocation-Table (nach jedem Login steht ein gesetztes Bit für einen belegten Block)
A91A und A91B A92A und A92B	A890 hex A8D0 hex	Zeiger auf DPB (Laufwerk A) Zeiger auf DPB (Laufwerk B)

Tabelle 3. Die wichtigsten Adressen des Disk-Parameter-Blocks für Laufwerk A (für Laufwerk B 64 Byte höher)

```
10 ' Disk-Parameterabfrage                           [EED2]
20 '                                                 [6052]
30 ' Oliver Suttorp, Johannisstr. 64                 [00E6]
40 ' 5024 Pulheim, Tel.: 02238/56368                 [817A]
50 '                                                 [6158]
60 OPENOUT "d":CLOSEOUT:'Login                       [1CF4]
70 aktdrive=PEEK(&A700):'aktuelles Drive
   holen                                             [1D1A]
80 'Anzahl der Blocks (formatabhaengig)              [65EC]
90 blocks=PEEK(&A895+aktdrive*64)                    [4FC0]
100 IF blocks=179 THEN format$="DATA"                [E6E6]
110 IF blocks=170 THEN format$="SYSTEM"              [266C]
120 IF blocks=155 THEN format$="IBM"                 [735A]
130 PRINT "Diskettenformat : ";format$               [FF76]
140 'Speicherplatzermittelung                        [26D4]
150 alltab=&A8B9+aktdrive*64:'Start des
    Block-Plans                                      [482E]
160 FOR byte=0 TO 21:'22 Bytes                       [0122]
170 wert=PEEK(alltab+byte)                           [9EF8]
180 anz=0                                            [A0DE]
190 FOR bit=0 TO 7                                   [A20E]
200 IF (wert AND (2^bit)) THEN anz=anz+1
                                                     [8382]
210 NEXT:belegt=belegt+anz:NEXT                      [ED78]
220 free=blocks+1-belegt                             [C51A]
230 PRINT "Von";blocks-1;"kB sind";free;
    "kB frei !"                                      [62F4]
```

Listing 4. Ein Programm zur Analyse des Disk-Parameterblocks

damit sind Sie noch immer auf die Standardformate System, DATA und IBM beschränkt. Andere (Fremd-)Formate können Sie noch nicht bearbeiten. Doch was ist überhaupt ein Fremdformat? Einfach ausgedrückt ist das ein Diskettenformat, das der CPC von Haus aus nicht erkennt, also auch nicht lesen oder schreiben kann. Demzufolge sind derartige Disketten auch mit normalen Kopierprogrammen wie »Disccopy« nicht zu kopieren. Damit ist auch die Hauptanwendung von Fremdformaten klar: Sie ergeben einen guten Kopierschutz. Um Programme gegen unerwünschtes Kopieren zu schützen, ist es ratsam, wichtige Teilprogramme in einem Fremdformat zu speichern. Diese Teile benötigen dann natürlich spezielle eigene Laderoutinen.

Zur Erzeugung eines »Fremdformats« muß die Diskette zunächst – wie jede Diskette – formatiert werden. Allerdings nicht mit dem Programm »Format« oder »Disckit« der Systemdiskette, sondern durch eine eigene Formatierroutine. Das Formatieren ist fast so einfach wie das Lesen und Schreiben eines Sektors. Es geschieht ebenfalls über einen »versteckten« RSX-Befehl. Der Befehl 86 hex verlangt im Prozessor-Register E die Laufwerksnummer und in D die Nummer der zu formatierenden Spur. Im C-Register teilen Sie der Routine den ersten Sektor mit, der auf die Spur geschrieben werden soll. Über das Doppelregister HL übergeben Sie nun einen Zeiger, der auf eine Tabelle mit den ID-Informationen deutet. In dieser Tabelle müssen die Werte aller Sektoren vermerkt sein, die später im jeweiligen ID-Feld stehen sollen. Unser Beispielprogramm in Listing 5 formatiert eine komplette Spur im Systemformat. Durch Änderung der IDs (»DEFB &41« bis »DEFB &49« hinter dem Label »IDTAB«) lassen sich nun fast beliebige Formate erzeugen. Beim ersten Blick auf das Beispielprogramm fällt Ihnen wahrscheinlich die ungewöhnliche Verteilung der Sektoren auf der Spur auf. Die Numerierung erfolgt nämlich nicht fortlaufend, sondern in Sprüngen von je fünf Werten. Prinzipiell ließen sich auch alle Sektoren nach IDs geordnet schreiben. Dadurch verlangsamen sich jedoch die Lese- und Schreibzugriffe, da der Computer zur Verarbeitung beziehungsweise Bereitstellung der Daten einige Zeit benötigt. Im ungünstigsten Fall müßte die Diskette beispielsweise eine gesamte Umdrehung vollziehen, ohne daß neue Daten gelesen oder geschrieben werden. Diese aus der Verarbeitungsgeschwindigkeit des Computers resultierende versetzte Verteilung der Sektoren einer Spur nennt man Skew-Faktor.

Mit Änderung der ID eines Sektors sind noch lange nicht alle Möglichkeiten der Fremdformatierung erschöpft. Wie wäre es beispielsweise mit Sektoren unterschiedlicher Länge? Bitteschön, das vierte Byte des ID-Felds gibt die Anzahl der Datenbyte pro Sektor an. Setzen wir dort anstelle der 2 eine 1, erfolgt die Formatierung mit Sektoren zu je 256 Byte. Das bedeutet natürlich, daß die doppelte Menge von Sektoren auf eine Spur paßt. 18 Sektoren zu je 256 Byte ersetzen nun die neun normalen 512-Byte-Sektoren. Mit solchen »Mini-Sektoren« läßt sich zum Beispiel recht einfach eine schnelle relative Dateiverwaltung aufbauen. Ein Datensatz entspricht dabei dem Inhalt eines Sektors. Benutzt man die letzten 30 Spuren der Diskette (Spur 10 bis 39) mit 18 Sektoren zu 256 Byte und formatiert die Spuren 0 bis 9 im DATA-Format, steht für das verwaltende Programm immer noch eine Kapazität von 38 KByte zur Verfügung. In die Spuren 10 bis 39 passen 540 Datensätze, die in Bruchteilen von Sekunden verfügbar sind, ohne den Arbeitsspeicher des CPC zu belasten.

Das Lesen und Schreiben solcher Sektoren ist etwas aufwendiger als das Formatieren. Mit dem Befehl 84 hex (Sektor lesen) wird nicht das gesamte ID-Feld an die Routine übergeben, sondern nur die Spur- und

```
; Formatieren einer einzelnen Spur in Maschinensprache
;
        LD      HL,BEFEHL   ; Adresse des Befehls
        CALL    &BCD4       ; Suche RSX-Befehl (KL-FIND-COMMAND)
        LD      (ADR),HL    ; Adresse speichern
        LD      A,C
        LD      (ROMNR),A   ; ROM-Nummer sichern (normal 7)
;
        LD      C,&41       ; erste Sektor-ID auf der Spur
        LD      D,0         ; Spurnummer nach D (0 bis 42)
        LD      E,0         ; Drive nach E (0=A; 1=B)
        LD      HL,IDTAB    ; Adresse der ID-Tabelle nach HL
        RST     &18         ; FAR-CALL zur gesuchten Adresse
DEFW    ADR                 ; Zeiger auf Adresse
        RET                 ; und zurück
ADR     DEFS    2,0         ; zwei Byte Speicher für die Adresse
ROMNR   DEFS    1,0         ; ein Byte Speicher für ROM-Nummer
;
IDTAB                       ; ID-Feld-Information    Sektor
        DEFB    0           ; Spurnummer              1
        DEFB    0           ; Kopfnummer              1
        DEFB    &41         ; Sektornummer            1
        DEFB    2           ; Sektorgröße (512 Byte)  1
        DEFB    0           ;                         2
        DEFB    0           ;                         2
        DEFB    &46         ;                         2
        DEFB    2           ;                         2
        DEFB    0           ;                         3
        DEFB    0           ;                         3
        DEFB    &42         ;                         3
```

Adresse (hex)	Normal (hex)	Bedeutung
A89F	41	Erster Sektor jeder Spur
A8A0	9	Anzahl der Sektoren pro Spur
A8A1	2A	Länge von GAP 3 beim Sektorlesen
A8A2	52	Länge von GAP 3 beim Formatieren
A8A3	E5	Füll-Byte beim Formatieren
A8A4	2	Byte pro Sektor (codiert: entspricht 512 Byte)
A8A5	4	Zahl der Records pro Sektor

Tabelle 4. Bei jedem Diskettenzugriff holt sich das DOS zunächst die Daten des FDC-Parameter-Blocks

Sektornummer. Die anderen Informationen für die Routine müssen also woanders stehen. Glücklicherweise hielten die Entwickler das Amsdos so flexibel, daß es keine festen Werte verwendet, sondern auf eine Tabelle im RAM zugreift, in der alle FDC-Parameter stehen. Aus diesem »FDC-Parameter-Block« beziehen alle DOS-Routinen ihre Werte. Tabelle 4 zeigt die Adressen mit den Werten der normalen Belegung. Daraus ist ersichtlich, daß die Größe eines Sektors auf Adresse A8A4 hex zu finden ist. Zum Lesen eines Sektors mit 256 Byte muß man also vor dem Aufruf der Routine 84 hex in diese Speicherstelle den Wert 1 laden. Das gleiche gilt analog für die Routine 85 hex (Sektor schreiben). Zwei Adressen des FDC-Parameterblocks sind auf den ersten Blick etwas unverständlich. Es sind die Adressen A8A1 und A8A2 hex, die der Angabe der Länge des Gap 3 dienen. Ein Gap ist der Leerraum zwischen den Sektorteilen (ID- und Daten-Feld) und den einzelnen Sektoren einer Spur. Dieser Leerraum gibt dem FDC Zeit, die gelesenen Daten (aus einem ID-Feld zum Beispiel) zu verarbeiten und an den Computer weiterzuleiten. Des weiteren dienen sie dazu, Gleichlaufschwankungen des Laufwerks auszugleichen. Dies ist besonders dann wichtig, wenn die Diskette nicht von dem Laufwerk beschrieben wurde, also zum Datenaustausch benutzt wird. Gap 3 beschreibt den Zwischenraum zweier Sektoren einer Spur. Durch geschickte Veränderung der Werte (Verkleinerung) in den genannten Adressen ist ein weiterer (zehnter) Sektor auf der Spur unterzubringen. Dabei ist lediglich zu berücksichtigen, daß zu kleine Werte Lesefehler zur Folge haben können. Besonders groß ist dieses Risiko, wenn die so behandelte Diskette später auf anderen Laufwerken zu laden sein soll.

```
        DEFB    2       ;       3
        DEFB    0       ;       4
        DEFB    0       ;       4
        DEFB    &47     ;       4
        DEFB    2       ;       4
        DEFB    0       ;       5
        DEFB    0       ;       5
        DEFB    &43     ;       5
        DEFB    2       ;       5
        DEFB    0       ;       6
        DEFB    0       ;       6
        DEFB    &48     ;       6
        DEFB    2       ;       6
        DEFB    0       ;       7
        DEFB    0       ;       7
        DEFB    &44     ;       7
        DEFB    2       ;       7
        DEFB    0       ;       8
        DEFB    0       ;       8
        DEFB    &49     ;       8
        DEFB    2       ;       8
        DEFB    0       ;       9
        DEFB    0       ;       9
        DEFB    &45     ;       9
        DEFB    2       ;       9
;
BEFEHL  DEFB    &86
```

Listing 5. Der Assembler-Quellcode eines Programms zur Formatierung einzelner Sektoren

```
; RSX-Befehlscode (Spur formatieren)
```

Die vorgestellten Tabellen im RAM sowie die ROM-Routinen erlauben bereits eine weitreichende Beeinflussung der Diskettenzugriffe. Aber immer noch ist der Rahmen des Machbaren längst nicht erschöpft.

Die jetzt folgende Stufe 4 der Disketten-Programmierung beschäftigt sich nur noch direkt mit dem FDC. Das DOS benötigen wir nun nicht mehr. Selbstverständlich ist die direkte Programmierung des Controllers sehr kompliziert, da man exakte Zeiten und Datenübergabe-»Zeremonien« einhalten muß. Für die exakte Einhaltung des Timing werden schon vom DOS sämtliche Interrupts bei Diskettenzugriffen gesperrt.

Da man schon in Stufe 3 mit relativ geringem Aufwand und mittleren Kenntnissen vielfältige Manipulationen vornehmen kann, wollen wir die Thematik der FDC-Programmierung jedoch nur der Vollständigkeit halber anschneiden. Der FDC ist nur über die beiden Portadressen FB7E und FB7F hex mit dem Z80-Prozessor des CPC verbunden. Eine dritte Adresse, FA7E hex, spricht ein Flip-Flop an, das die Laufwerksmotore ein- beziehungsweise ausschaltet. Der Datentransfer zwischen Computer und Controller ist im CPC so realisiert, daß der Computer immer anhand des Haupt-Statusregisters feststellt, ob der FDC zum Datenaustausch bereit ist. Das ist zwar die einfachste Art des Datenaustausches, hat aber den Nachteil, daß sich der Computer während dieser Zeit um nichts anderes kümmern kann. Der Diskettenzugriff hat also absoluten Vorrang. Der FDC ist aber für zwei weitere Betriebsarten vorbereitet. Als erstes ist der DMA-Modus (Direct-Memory-Access) zu nennen. In diesem Modus hat der FDC in Zusammenarbeit mit einem DMA-Controller (findet in vielen Festplatten-Laufwerken Verwendung) direkten Zugriff auf den Arbeitsspeicher des Computers, das heißt, der Hauptprozessor ist dann von der Aufgabe entbunden, die Daten zu holen oder bereitzustellen. Das führt natürlich zu enormen Geschwindigkeitsvorteilen. Der zusätzliche Hardware-Aufwand eines DMA-Controllers hätte sich bei der Entwicklung der CPCs aber allein des Diskettenlaufwerks wegen kaum gelohnt, so daß wir heute auf diesen Komfort leider verzichten müssen.

Der zweite Modus ist der Interrupt-Modus. Dabei signalisiert der FDC der CPU mit einem Interrupt seine Bereitschaft zum Datentransfer. Auf diese Betriebsart wurde in den CPCs wahrscheinlich verzichtet, da das CPC-Betriebssystem mit einer Vielzahl von Interrupts ohnehin schon an der Grenze seiner Leistungsfähigkeit betrieben wird.

Die ganze Programmierung des FDC beschränkt sich also auf zwei (drei) Adressen. Das Haupt-Statusregister ist mit dem Flag-Register des Z80 vergleichbar (Tabelle 5). Aus ihm lassen sich nur Daten lesen. Das Datenregister dagegen steht für Lese- und Schreibzugriffe zur Verfügung. Über das Datenregister erfolgt deshalb auch die Programmierung. Als Beispiel für die FDC-Programmierung dient ein Kommando, dessen Auswirkung aus den unteren drei Programmierungsstufen nicht erreichbar ist: das Lesen einer Sektor-ID. Bevor Sie mit der FDC-Programmierung beginnen, stellen Sie anhand des Haupt-Statusregisters fest, ob der Controller momentan zur Befehlsverarbeitung bereit ist. Danach schalten Sie die Laufwerks-Motore mit einem OUT-Befehl ein. Die eigentliche Programmierung ist in drei Phasen unterteilt: die Befehlsphase, die Ausführungsphase und die Ergebnisphase.

Als erstes wird dem FDC in der Befehlsphase der Kommando-Code 4A hex für das Lesen der ID übermittelt. Danach folgt ein Datenbyte, das dem FDC sagt, auf welches Laufwerk sich der Befehl bezieht. Für Laufwerk A schreiben Sie eine 1, für Laufwerk B

eine 2 in das Datenregister. Damit ist die Befehlsphase schon abgeschlossen. Zwischen den Datenübergaben dürfen Sie aber nicht vergessen, im Haupt-Statusregister abzufragen, ob der FDC empfangsbereit ist. In der jetzt folgenden Ausführungsphase erledigt der FDC seine Aufgabe; er sucht auf der aktuellen Spur nach einer Sektor-ID. Die erste gefundene liest er in sein Register ein. Der Computer muß in der Ausführungsphase keine Daten vom FDC holen. Das ist allerdings nur bei diesem Befehl so. Bei anderen, wie beispielsweise »Sektor lesen« oder »Sektor schreiben«, werden alle Datenbyte (normalerweise 512 an der Zahl) in der Ausführungsphase zwischen CPU und FDC ausgetauscht. Nachdem jetzt mit der Ausführungsphase die zweite Phase endet, beginnt augenblicklich die dritte und letzte Phase. In dieser Ergebnisphase werden zuerst die drei internen Statusregister 0 bis 2 für den CPC zur Verfügung gestellt. Das Statusregister 3 läßt sich nur über einen bestimmten Befehl auslesen. Es beinhaltet den Laufwerksstatus, sagt also aus, ob das Laufwerk betriebsbereit oder der Schreibschutz aktiv ist. Aus diesen Registern kann der Computer eventuelle Fehler oder Hardware-Eigenschaften der Diskettenlaufwerke abfragen. Auf diese drei Statusbyte folgen die erfragten vier Byte der Sektor-ID. Nacheinander stellt der FDC Spurnummer, Kopfadresse, Sektornummer und Sektorgröße bereit. In der Ergebnisphase ist zu beachten, daß immer alle Byte ausgelesen werden müssen, auch wenn man sie nicht benötigt. Der FDC kann einen neuen Befehl nur dann ausführen, wenn von der Ausführung des vorangegangenen Befehls keine Daten mehr übrig sind.

Wenn wir jetzt Ihren Appetit auf die FDC-Programmierung angeregt haben, können Sie die Aufstellung der gesamten Befehle in einem Datenblatt des FDC nachlesen. Die Dokumentation aller Befehle würde den Rahmen dieses Artikels bei weitem sprengen. Es ist auch sinnvoll, ein gut dokumentiertes DOS-Listing anzuschaffen.

Doch – warum sollten Sie sich mit einer solchen Programmierung überhaupt herumschlagen? Für den normalen Anwender, der nur Programme und Daten von Diskette laden will, ist es sicher uninteressant; ihm genügen die DOS-Befehle allemal. Aber für jemanden, der verstehen möchte, was im »Hintergrund« passiert, ist es höchst interessant, das Laufwerk »nach seiner Pfeife tanzen zu lassen«. Und die Handhabung des FDC ist auf größtmögliche Benutzerfreundlichkeit ausgelegt, was dazu führt, daß wenige leistungsstarke Befehle das Leben des Programmierers erleichtern. Durch die zusätzlichen Fähigkeiten, wie unterschiedliche Sektorgröße und schnelle Laderoutinen, kann man jetzt eigenen Diskettenprogrammen höheren Komfort und eine höhere Arbeitsgeschwindigkeit angedeihen lassen.

Last but not least wollen wir aber auch die Hauptanwendung für solche »Bit-Verrenkungen« nicht verschweigen: Kopierschutzmechanismen für mehr oder weniger teure Programme. Sie sind durch die diversen Befehle des FDC ohne großen Aufwand zu realisieren. Aber genau da ist auch der Haken an der Sache: Durch die begrenzte Anzahl von Befehlen ist es unmöglich, einen Schutz zu verwirklichen, der sich nicht von einem ebenso intelligenten Kopierprogramm kopieren ließe. Mehr über dieses Thema lesen Sie im Anschluß an diesen Beitrag.

(Oliver Suttorp/ja)

Literaturhinweis:
Günter Woigk, »Das Schneider CPC Systembuch«, Sybex Verlag, ISBN 3-88745-606-8, Preis: 58 Mark
Brückmann/Schieb, »Das große Floppybuch zum CPC«, Data Becker, ISBN 3-89011-093-2, Preis: 49 Mark
»DDI-1 Firmware Manual«, Schneider-Data, Am Rindermarkt 4a, 8050 Freising, Preis: 59 Mark (englische Originalausgabe)

Bit	Mnemonic	Bedeutung
0 bis 3	DB	Jedes Bit steht für eins der vier möglichen Laufwerke. Ist das entsprechende Bit gesetzt, kann auf dem Laufwerk ein Schreib- oder Lesebefehl ausgeführt werden.
4	CB	Dieses Bit signalisiert, daß der FDC zur Zeit ein Kommando abarbeitet.
5	EXM	Mit diesem Bit kann zwischen den einzelnen Phasen des FDC unterschieden werden. In der Resultatphase ist das Bit 0.
6	DIO	Hiermit bestimmt der FDC, ob er bereit ist, Daten zu senden (1) oder zu empfangen (0).
7	RQM	Ein gesetztes Bit zeigt an, daß der FDC bereit ist, Daten mit dem Prozessor auszutauschen. Die Richtung bestimmt Bit 6.

Tabelle 5. Das Hauptstatusregister des FDC 765 (Port FB7E hex)

Bit	Mnemonic	Bedeutung
0 und 1	US	Diese Bits geben an, welches Laufwerk angesprochen wurde.
2	HD	Der gewählte Kopf des Laufwerks (immer 0).
3	NR	Das angesprochene Laufwerk war nicht bereit.
4	EC	Das Laufwerk hat einen Fehler gemeldet.
5	SE	Hat der FDC eine bestimmte Spur (Seek-Befehl) gefunden, setzt er dieses Bit.
6 und 7	IC	Diese Bits setzt der FDC beim Auftreten von Fehlern.

Das Statusregister 0

Bit	Mnemonic	Bedeutung
0	MA	keine Sektor-ID auf der Spur gefunden
1	NW	die Diskette ist schreibgeschützt
2	ND	der FDC setzt dieses Bit, wenn er beim Schreiben oder Lesen den Sektor nicht fand
3	-	immer 0
4	OR	ein Datenbyte wurde nicht früh genug vom FDC geholt
5	DE	es ist ein Prüfsummenfehler entstanden (die Prüfsumme generiert der FDC)
6	-	immer 0
7	EN	das Ende der Spur ist erreicht

Das Statusregister 1

Bit	Mnemonic	Bedeutung
0	MD	Data-Adress-Mark wurde nicht gefunden
1	BC	Die physikalische Spurnummer stimmt nicht mit der in der ID überein
2	SN	gesuchter Sektor nicht gefunden
3	SH	Sektor mit gleicher ID gefunden
4	WC	siehe Bit 1
5	DD	Prüfsummenfehler im Datenfeld (siehe Register 1, Bit 5)
6	CM	Sektor mit gelöschter Data-Adress-Mark gefunden
7	-	immer 0

Das Statusregister 2

Bit	Mnemonic	Bedeutung
0 und 1	US	angemeldetes Laufwerk
2	HD	gewählter Kopf des Laufwerks
3	TS	Ein gesetztes Bit zeigt an, daß ein Doppelkopflaufwerk angeschlossen ist
4	T0	Kopf des aktiven Laufwerks ist auf Spur 0
5	RY	das gewählte Laufwerk ist betriebsbereit
6	WP	die eingelegte Diskette ist schreibgeschützt
7	FT	im Laufwerk ist ein Fehler aufgetreten

Das Statusregister 3

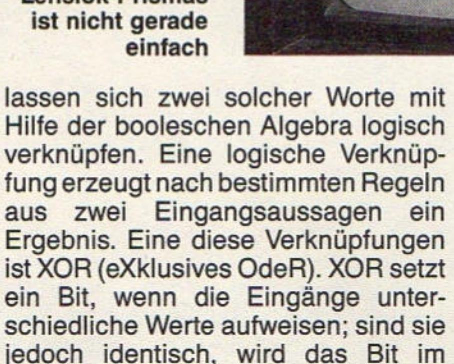
Schutz – wie lange noch?

Wir sind noch nie vor »heißen Eisen« zurückgeschreckt. Auch ein Beitrag wie dieser, der sich mit Grundlagen von Kopierschutz-Methoden beschäftigt, birgt natürlich die Gefahr des Mißbrauchs.

Kopierschutz – schon das Wort übt eine fast magische Faszination auf viele Computerbesitzer aus. Sie verstehen nicht, wie so etwas funktioniert und betrachten den Begriff, als käme er aus einer anderen Welt. Den Leuten, die sich damit auseinandersetzen, im allgemeinen als »Cracker« bezeichnet (nicht jedoch »Hacker«), haftet ein zweifelhafter Ruf an, der sie an die Grenze der Illegalität oder manchmal sogar darüber hinaus bringt. Dieser Ruf wird von vielen Softwarehäusern geschürt, da sie um die Verkaufszahlen ihrer Programme fürchten. Dabei wird meistens jedoch übersehen, daß der durchschnittliche Cracker nicht die Absicht verfolgt, Softwarehäuser in den Bankrott zu treiben. Vielmehr will er meist sein eigenes Können an dem des Kopierschutz-Programmierers messen und so einen privaten Wettkampf ausfechten. Wir wollen mit diesem Beitrag auch keineswegs Rezepte zum Entfernen des Kopierschutzes geben, sondern Techniken aufzeigen, die heute zum Schutz von Programmen Verwendung finden. Ähnlichkeiten mit zur Zeit benutzten Kopierschutzmechanismen sind natürlich nicht zu vermeiden.

Schon unter Basic gibt es einige Wege, Fremden die Einsicht in ein Programm zu verweigern. Mit einem »POKE &172,0« läßt sich der ersten Programmzeile die Nummer 0 geben. Dieser Eingriff hat zur Folge, daß diese Zeile nicht mehr zu listen und zu editieren ist, sehr wohl aber ausgeführt wird. In dieser Zeile bringt man beispielsweise eine Befehlsfolge unter, die eine Binärdatei lädt und startet. Um Kopierwütige zu verwirren, sollte man in den sichtbaren Zeilen ein von der Logik her nicht funktionsfähiges Programm ablegen. Der POKE nutzt eine Lücke des Basic-Interpreters, da dieser für Zeilen mit der Nummer 0 eigentlich gar nicht gerüstet ist. Genau so einfach, wie dieser »Schutz« zu erzeugen ist, läßt er sich aber auch wieder entfernen. Der »POKE &172,zeilennummer« gibt der Zeile 0 nämlich die Nummer <zeilennummer>, und wer es noch einfacher haben will, gibt nur den Befehl RENUM ein.

Die meisten Schutzmechanismen basieren aber darauf, daß eine Laderoutine das Hauptprogramm nachlädt. Dieser wichtigste Programmteil



```
Bitte geben Sie einen Satz ein      :? Happy Computer
Bitte geben Sie das Codierwort ein :? Computer-Spass

Text         : H  a  p  p  y     C  o  m  p  u  t  e  r
ASCII-Werte  : 48 61 70 70 79 20 43 6F 6D 70 75 74 65 72

XOR

Code         : C  o  m  p  u  t  e  r  -  S  p  a  s  s
ASCII-Werte  : 43 6F 6D 70 75 74 65 72 2D 53 70 61 73 73

Ergebnis     : ↑           T  &     @  #     ×
ASCII-Werte  : 0B 0E 1D 00 0C 54 26 1D 40 23 05 15 16 01
```

Bild 2. Auch Texte lassen sich mit XOR bis zur Unkenntlichkeit chiffrieren

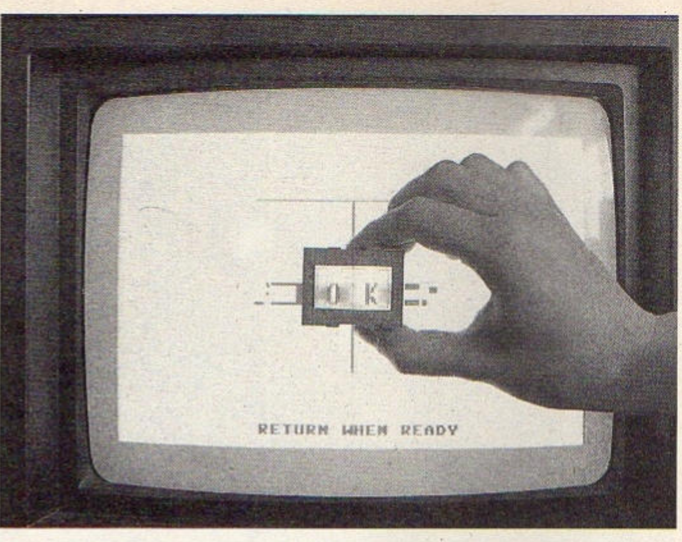


Bild 1. Die Handhabung des Lenslok-Prismas ist nicht gerade einfach

jedoch ist in keinem systemkonformen Dateiformat abgelegt. Speziell bei Diskettensoftware wird oft zusätzlich eine fehlerhafte Spur abgefragt, die in herkömmlicher Weise nicht zu kopieren ist. Der Schwachpunkt jedes dieser Schutzsysteme ist das Ladeprogramm, da es im normalen Format vorliegen muß. Schafft es ein Cracker, in diesem Ladeprogramm hinter der Laderoutine eine eigene Routine zur Speicherung einzubinden, ist der Schutz meist hinfällig. Dieser Zugang läßt sich auf einigen Wegen mehr oder weniger stark erschweren. Da in Maschinensprache viel mehr Schutzmethoden denk- und realisierbar sind, werden nur die wenigsten Lader in Basic geschrieben. Um zu verhindern, daß der Cracker ein Ladeprogramm zu leicht durchschaut, sind solche Routinen oft mit speziellen Verfahren verschlüsselt. Dafür wurden verschiedene Methoden entwickelt, die ein relativ hohes Maß an Sicherheit bieten. Darunter fällt zum Beispiel ein System, das vor einiger Zeit in den USA entstand und sich recht gut bewährte. Es benutzt zwei »Schlüssel«. Als Schlüssel dienen Zahlen oder Buchstabenfolgen, mit denen man einen Text codiert beziehungsweise wieder decodiert. Genauso wie man mit einem Türschlüssel eine Tür öffnet, öffnet solch ein Schlüssel den Zugang zu den Daten. Einer der Schlüssel ist dem Benutzer bekannt, es handelt sich um den sogenannten Benutzerschlüssel. Den zweiten Schlüssel kennt der Empfänger der Daten (das Ladeprogramm im Arbeitsspeicher des Computers), da er nur mit ihm die Daten wieder entschlüsseln kann. Natürlich eignen sich solche Verfahren nur bedingt für die Codierung von Programmen. Hier macht man sich die Arbeitsweise des Computers zunutze. Die CPU unterscheidet nur zwischen zwei Signalen beziehungsweise Zuständen (Ja/nein, wahr/falsch, 1/0 oder Strom/kein Strom). Da jedes Datenwort (Byte) aus jeweils acht dieser Zustände besteht, lassen sich zwei solcher Worte mit Hilfe der booleschen Algebra logisch verknüpfen. Eine logische Verknüpfung erzeugt nach bestimmten Regeln aus zwei Eingangsaussagen ein Ergebnis. Eine diese Verknüpfungen ist XOR (eXklusives OdeR). XOR setzt ein Bit, wenn die Eingänge unterschiedliche Werte aufweisen; sind sie jedoch identisch, wird das Bit im Ergebnis null.

Beispiel 1:

```
         10100010
XOR      01110100
ergibt   11010110
```

Beispiel 2:

```
         11010110
XOR      01110100
ergibt   10100010
```

Diese Beispiele verdeutlichen das Besondere an dieser Verknüpfung: Behandelt man das Ergebnis der ersten Verknüpfung nochmals mit dem gleichen Wert, ergibt sich wieder der Ausgangswert. Den Zeitaufwand, den die Entschlüsselung eines Programms verursacht, selbst wenn es einen großen Teil des Speichers belegt, bemerkt der Benutzer kaum. Solche Entschlüsselungs-Routinen arbeiten nämlich meist in Maschinencode und verarbeiten mehrere tausend Byte pro Sekunde. Bei der Verwendung von XOR-Schleifen sind der Fantasie fast keine Grenzen gesetzt. So lassen sich selbstverständlich mehrere Verknüpfungen mit unterschiedlichen Werten hintereinander aufrufen oder gar ineinander verschachteln. Geben Sie im Beispielprogramm (Listing) ein beliebiges Wort Ihrer Wahl ein. Das Ergebnis ist ohne Kenntnis des Codewortes kaum zu erraten (siehe auch Bild 1). Jetzt geben Sie in das Beispielprogramm einmal ein Codewort aus lauter gleichen Buchstaben ein. Am Ergebnis werden Sie den Nachteil dieser Methode erkennen: Gleiche Werte sind auch nach der Verknüpfung gleich.

In vielen Programmen kommen Null-Bytes am häufigsten vor. Findet man also eine Häufung gleicher Bytes, handelt es sich also wahrscheinlich um Nullen und man kann an ihnen unschwer »erraten«, mit welchem Wert das Programm verschlüsselt ist.

Ähnliche Verfahren finden trotzdem als Kopierschutz Verwendung. Zur Bildung des Schlüsselwerts benutzt man aber kein festes, im Programmcode enthaltenes Byte. Man nimmt vielmehr das Refresh-Register des Z80 zur Hilfe, das sich nach jedem Taktzyklus des Z80 um den Betrag 1 erhöht (der Zyklus entspricht dem Arbeitstakt der CPU). Lädt man zunächst einen festen Wert in dieses Register und liest es nach einer genau definierten Zeit wieder aus, erhält man einen veränderten Wert, der ausschließlich von der Laufzeit des Programms abhängig ist. Jede noch so geringe Veränderung des Programms hat zwangsläufig Einfluß auf die Anzahl der benötigten Takt-Zyklen. Nutzt man nun den so ermittelten Wert zur Decodierung, läßt das zwangsläufig falsch entschlüsselte Programm den Computer augenblicklich »abstürzen«. Das Ziel, jede Veränderung im Programm zu unterbinden und somit den Kopierschutz zu erhalten, ist mit diesem System fast erreicht. Aber natürlich ist auch dieser Schutz nicht vollkommen, er erschwert Crackern lediglich ihr Handwerk. Man kann beispielsweise in der zeitabhängigen Schleife Befehle gegen andere austauschen, die genausoviel Zeit benötigen, um so die Laufzeit nicht zu verändern, das Programm aber nach seinen Wünschen umschreiben.

Dieses Verfahren wurde beim CPC erstmals im »Speedlock-Protection-System« für Kassetten eingesetzt und fand durch seine neue Verfahrensweise recht weite Verbreitung. Doch gerade in der gemeinsamen Nutzung

Markt&Technik
68000er
DAS MAGAZIN DER NEUEN COMPUTER-GENERATION
FEBRUAR 1987
2/87
DM 7,-
Atari ST
Amiga
Grafik-Sensation
Deluxe Paint II
QL
Große Assembler-Parade
Mac
kreativ!
Interview mit »Axel F.« – Komponist Harold Faltermeyer
Viele anspruchsvolle Listings für Atari ST, Amiga und QL
AUFBRUCH IN EINE NEUE DIMENSION
mit »68000er«, dem Magazin der neuen Computer-Generation
▶ Programmiersprachekurse für Basic, C, Modula und Assembler.
▶ Bauanleitungen für professionelle Hardware-Erweiterungen.
▶ Spiele-Spaß und -Spannung auf höchstem Niveau.
Ihre hot-line zur Spitzentechnologie von Atari ST, Amiga, Macintosh und Sinclair QL.
Das »68000er«-Magazin erscheint
jeden Monat neu!
POSTER & GUTSCHEIN
KOSTENLOS FÜR SIE
84 mal 60 Zentimeter High-Tech-Szene erwarten Sie! Ihr »68000er«-Poster ist im Abonnementpreis enthalten und gehört Ihnen, auch wenn Sie Ihre Bestellung widerrufen sollten.
FÜR EIN KOSTENLOSES PROBEEXEMPLAR DES »68000er«-MAGAZINS
JA, ich möchte »68000er«, das Magazin der neuen Computer-Generation, kennenlernen.
Senden Sie mir bitte die aktuellste Ausgabe kostenlos als Probeexemplar. Wenn mir »68000er« gefällt und ich es regelmäßig weiterbeziehen möchte, brauche ich nichts zu tun: Ich erhalte es dann regelmäßig frei Haus per Post. Außerdem nutze ich den Abonnement-Preisvorteil von 8% und bezahle pro Jahr nur 77,– DM statt 84,– DM im Einzelverkauf.
Vorname
Name
Straße
PLZ, Ort
Datum 1. Unterschrift
Mir ist bekannt, daß ich diese Bestellung innerhalb von 8 Tagen bei der Bestelladresse widerrufen kann und bestätige dies durch meine zweite Unterschrift. Zur Wahrung der Frist genügt die rechtzeitige Absendung des Widerrufs.
Datum 2. Unterschrift
Gutschein ausfüllen und absenden an: Markt&Technik Verlag Aktiengesellschaft, Vertrieb, Postfach 1304, 8013 Haar
HCS18
WWW.HOMECOMPUTERWORLD.COM

eines Schutzes liegt auch eine große Gefahr: Ist er erst einmal geknackt, betrifft das nicht nur ein Programm – vielmehr sind auf einen Schlag gleich ganze Software-Serien den Raubkopierern schutzlos ausgeliefert. So erging es einem Softwarehaus, das sein ganzes Vertrauen in Speedlock setzte. Fast alle Programme des Labels waren mit diesem Lader versehen. Deshalb gibt es seit längerem einige käufliche Kopierprogramme, für Speedlock-geschützte Kassetten. Da es aber inzwischen als Reaktion darauf diverse Variationen des Speedlock gibt, arbeiten diese Kopierprogramme nur bei einem Teil korrekt.

Einfache Direktkopien

Eine äußerst einfache, aber oft genauso wirkungsvolle Kopiermethode für Kassetten-Software besteht darin, zwei Kassettenrecorder direkt miteinander zu koppeln und so eine identische Kopie vom Orginalband zu erhalten. Dabei wird der Kopierschutz mitkopiert und braucht so nicht entfernt zu werden. Auch diesen Praktiken arbeiteten die Schutz-Entwickler entgegen. Auch Speedlock verwendet einen Schnell-Lader (daher sein Name), der zwar eine Direktkopie vom Originalband nicht verhindern kann, der jedoch oft gewährleistet, daß durch beim Überspielen unvermeidliche Qualitätsminderungen Kopien nicht fehlerfrei lesbar sind.

Andere Autoren kamen auf weniger elegante Lösungen. Das Lenslok-System beispielsweise basiert weniger auf einem softwaremäßigen Schutz (der allerdings in Form eines Turboladers zusätzlich vorhanden ist), als mehr auf einer Hardwarelösung: Dies sind in diesem Falle spezielle Prismen, durch die man auf dem Bildschirm eine Buchstabenkombination erkennt (Bild 2). Diese Buchstaben sind bei jedem Start andere und müssen als Codewort via Tastatur eingegeben werden. Dafür stehen dem Benutzer drei Versuche zur Verfügung. Schafft er es nicht, die korrekte Kombination einzugeben, löst das Programm einen Reset des Computers aus.

Selbst wenn man es schafft, das Programm zu kopieren, läßt es sich ohne die »Hardware-Erweiterung« (das Prisma) nicht nutzen. Einen schwerwiegenden Nachteil stellt die extreme Benutzerunfreundlichkeit dar. Erstens gibt es zu verschiedenen Programmen natürlich auch verschiedene Prismen. Hat man also aus der »Glassammlung« nach zahlreichen erfolglosen Versuchen (und ebenso vielen erneuten, quälend langen Ladevorgängen) endlich das passende Prisma herausgesucht, steht man unvermittelt vor der zweiten Hürde. Die Prismen sind qualitativ teilweise so schlecht gefertigt (sie werden aus Kostengründen nicht geschliffen sondern aus Kunststoff gegossen), daß man – wenn überhaupt – die beiden Buchstaben nur mit Schwierigkeiten erkennt. Dieses System ist für beide Datenträger (Kassette und Diskette) erhältlich, während Speedlock zunächst nur Kassettensoftware zu schützen trachtete. Inzwischen haben die Speedlock-Entwickler, D. Looker und Aubrey Jones, auch eine Diskettenversion ihres Cracker-Schrecks herausgebracht. Es basiert auf einer einfachen Fehlerspur-Abfrage. Die gesamte Diskette ist in einem normalen Format gehalten. Selbst die Programmteile liegen als normale Dateien vor. Nur das Hauptprogramm ist mit einem Wert über die XOR-Verknüpfung verschlüsselt. Speedlock selbst besteht aus zwei Dateien, der Laderoutine und einer Datei mit dem sinnesschwangeren Namen »AAAA.BIN«. Die Laderoutine initialisiert die Farben und aktiviert das Controller-ROM. Danach lädt es den zweiten Teil des Kopierschutzes, eben jene Datei »AAAA.BIN«. Der eigentliche »Witz« ist, daß die XOR-Schleife zur Entschlüsselung des zweiten Teils in keiner der beiden Binär-Dateien zu finden ist. Hier haben sich die Entwickler etwas Besonderes einfallen lassen. Doch zur Erklärung müssen wir etwas weiter ausholen.

Routine im Header

Aus Kompatibilitätsgründen haben die Programmierer des DOS den Datei-Header des Kassetten-Managers ohne Änderung übernommen. Dieser Header geht den Daten einer Datei voran. In ihm sind unter anderem Lade-, End- und Startadresse gespeichert, einige Bytes jedoch bleiben ungenutzt (umfassende Informationen darüber vermittelt Ihnen unser Beitrag auf Seite 31 dieser Ausgabe). Da der Header ab Adresse A750 hex in den Arbeitsspeicher geladen wird, haben die Speedlock-Entwickler ihre XOR-Routine dort plaziert. Genau gesagt liegt die Routine im »Benutzer-Feld«, das von A771 bis A794 hex reicht und dem Anwender bei der Speicherung zur freien Verfügung steht. Diese Routine entschlüsselt die Datei »AAAA.BIN« und startet danach das darin enthaltene Programm. Um Cracker, die so weit vorgedrungen sind, durch Demotivation zur Aufgabe ihres unschönen Vorhabens zu bewegen, befindet sich dort eine weitere, zeitabhängige Schleife. Sie entschlüsselt diesen Programmteil ein zweites Mal, nun aber abhängig vom Ergebnis der vorangegangenen Entschlüsselung. Danach erfolgt die Abfrage der fehlerhaften Spur und das Hauptprogramm wird geladen, wiederum entschlüsselt und endlich gestartet.

Ein weiterer diskettengestützter Kopierschutz ist der P.M.S.-Lader, der in jüngster Zeit bei einigen Programmen Verwendung findet. Er greift nicht auf Dateien zu, sondern lädt alle Daten, wie Titelbild und Hauptprogramm, direkt aus den Diskettensektoren. Der Start des Ladeprogramms erfolgt oft mit dem Basic-RSX-Befehl ICPM, was keineswegs das Booten des Betriebssystems CP/M zur Folge hat. Der Befehl startet in diesem Fall nur den Lader auf etwas ungewöhnliche Weise. Dieses Maschinencode-Programm darf höchstens 512 Byte lang sein und muß im ersten Sektor (41 hex) der Spur 0 liegen. Bei Ausführung des RSX-Befehls ICPM geschieht nichts anderes, als daß das Betriebssystem diesen ersten Sektor der Spur 0 nach Adresse 100 hex lädt und anschließend die Kontrolle an diese Adresse übergibt. Damit läuft nun bereits die Laderoutine und lädt die einzelnen Teile des Hauptprogramms aus den Sektoren. Die Diskette liegt in einem relativ gebräuchlichen Format vor, das heißt jede Spur umfaßt neun Sektoren zu je 512 Byte. Der P.M.S.-Lader verwendet aber nur acht dieser neun Sektoren als Datenspeicher, pro Spur sind also nur vier KByte des Programms gespeichert. Der neunte Sektor jeder Spur ist in zwei Hälften zu je 256 Byte aufgeteilt. Die erste Hälfte belegt jeweils eine kleine Routine, die der Lader mit einem CALL als Unterprogramm aufruft. Die Routinen erledigen diverse Aufgaben. Sie setzen zum Beispiel die Farben oder bauen den Bildschirm – für P.M.S. charakteristisch – von oben nach unten auf. Die zweite Hälfte dient als Adressenspeicher. Sie enthält, natürlich verschlüsselt, die Adressen für den zu benutzenden Speicherbereich und die Startadresse der Hilfsroutinen. Darüber hinaus findet der Lader hier die Werte zur Decodierung des nächsten Datensektors.

Damit aber noch nicht genug der kleinen »Gemeinheiten«. Nachdem das Laden des gesamten Programms abgeschlossen ist, startet der Lader erst eine »Haupt-Test-und-Start-Routine«, die er auf der letzten Spur

```
10 ' Verschluesselung mittels der            [60EE]
20 '  logischen Operation   XOR              [1E9E]
30 '                                         [5F54]
40 ' Oliver Suttorp, Johannisstr. 64         [6DE8]
50 ' 5024 Pulheim, Tel.: 02238/56368         [E27C]
60 '                                         [845A]
70 MODE 2                                    [FBFC]
80 INPUT "Bitte geben Sie einen Satz ein
   <5>:";text$                               [4842]
90 INPUT "Bitte geben Sie das Codierwort
    ein :";code$                             [2242]
100 x=LEN(code$):z=LEN(text$):IF z>20 TH
    EN 70                                    [8D70]
110 LOCATE 1,5:PRINT "Text<8>: ";            [7302]
120 stg$=text$:GOSUB 270:GOSUB 230           [335A]
130 PRINT:PRINT"XOR":PRINT:PRINT"Code<8>
    : ";                                     [EC4E]
140 FOR i=0 TO z/x:c$=c$+code$:NEXT:code
    $=c$                                     [9376]
150 stg$=code$:GOSUB 270:GOSUB 230           [3A0C]
160 PRINT:PRINT"Ergebnis<4>: ";              [B826]
170 ' *** Satz verschluesseln                [C8AA]
180 FOR i=1 TO z:erg$=erg$+CHR$(ASC(MID$
    (text$,i,1)) XOR ASC(MID$(code$,i,1)
    ))                                       [24BE]
190 NEXT                                     [75F2]
200 stg$=erg$:GOSUB 270:GOSUB 230            [1A4A]
210 CALL &BB06:RUN                           [905C]
220 ' *** ASCII-Werte in Hex ausgeben        [D8B8]
230 PRINT:PRINT"ASCII-Werte : ";             [713E]
240 FOR i=1 TO z:PRINT HEX$(ASC(MID$(stg
    $,i,1)),2)" ";:NEXT                      [1406]
250 PRINT:RETURN                             [48BC]
260 ' *** Buchstaben ausgeben                [6C0C]
270 FOR i=1 TO z:PRINT CHR$(1);MID$(stg$
    ,i,1);"<2>";:NEXT                        [857E]
280 RETURN                                   [1734]
```

Listing. Mit diesem kurzen Beispiellisting können Sie ein wenig zur Probe codieren

findet. Dieses Programm fragt eine bestimmte fehlerhafte Spur der Diskette ab und fabriziert, wenn diese nicht auffindbar ist, einen farbenfrohen »Absturz«. Ist auch diese letzte Hürde überwunden, startet das (Spiel-)Programm, das nach so vielen Mühen nun endlich richtig im Speicher steht.

Allein das Vorhandensein dieses letzten Programmteils mit seinem abschließenden Test zeigt, daß die Programmierer eines Kopierschutzes dem Erfolg ihrer Bemühungen nie voll vertrauen.

Beispiele für Schutzmechanismen, die ihre Aufgabe nicht ganz erfüllen, sind zwar durch intensive Programmkontrollen seitens der Softwarehäuser rar geworden, ab und zu taucht aber doch einmal wieder einer auf und verursacht in der Regel ein hämisches Grinsen der Kopierwilligen.

Interessant ist es, die Entwicklung von Schutzmechanismen bei Diskettensoftware im Laufe der Jahre zu verfolgen. Begonnen hat alles mit einfachen Veränderungen der Sektor-IDs. Eine einzelne Sektornummer wird dabei auf der gesamten Diskette geändert. Unter anderem arbeitet ein bekanntes Textverarbeitungsprogramm nach diesem Prinzip. In diesem Sektor befindet sich eine kleine Maschinencode-Routine, die das Hauptprogramm bei jeder Diskettenoperation aufruft. Findet das Programm diese Routine nicht, liegt demzufolge eine »Raubkopie« vor und es beginnt, die Daten der Diskette nach und nach zu zerstören. Der Vorteil dieser subtilen »Langzeitwirkung« liegt darin, daß sich eine vermeintlich einwandfreie Kopie erst nach einiger Zeit als unbrauchbar erweist.

Die Entwickler anderer Disketten-Software beließen es nicht bei nur einem Sektor. Praktisch die ganze Diskette ist in einem »Fremdformat« beschrieben. Dabei ist der Fantasie bei Vergabe von Sektornummern kaum eine Grenze gesetzt. Ob sie nun von F1 bis F9 hex oder von 10 bis 18 hex numeriert sind, ist vollkommen egal. Nur Nummern der normalen Formate scheiden aus, weil Kopien sonst mit einfachsten Mitteln herzustellen sind.

Fast gleichzeitig wurde entdeckt, daß man mehr als 40 Spuren auf einer Diskette unterbringen kann. Auf den 3-Zoll-Disketten lassen sich bis zu 43 Spuren beschreiben. Da einfache Kopierprogramme, wie die mit dem Laufwerk gelieferten Disckit oder Disccopy nur die Spuren 0 bis 39 erkennen und verarbeiten, konnte man in diese »verborgenen« Spuren wichtige Teile des Programms auslagern.

In diesem Zeitraum kam die Disk-Utility »Oddjob« von Pride Utilities auf den Markt. Dieses Softwarepaket enthielt unter anderem ein Kopierprogramm, mit dem sich auch Formate mit »illegalen« Sektornummern duplizieren ließen. Nur Spuren jenseits der Nummer 39 waren auch für Oddjob ein »Rätsel«. Dieses Manko behoben findige Cracker jedoch recht bald.

Rüstungsspirale

Die Softwareschützer waren also wieder gefordert. Sie mußten sich im Wettlauf mit den Herstellern von Kopierprogrammen etwas Neues einfallen lassen. So wurden kurz darauf Sektoren mit größeren Kapazitäten als den üblichen 512 Byte verwendet: Bis zu 4096 Byte paßten auf einmal in einen Sektor.

All diesen Mechanismen wohnt ein großer Vorteil inne. Bei der Entwicklung des Schutzes konnten die Programmierer trotz der ungewöhnlichen Formate auf Betriebssystemroutinen zurückgreifen und mußten sich so nicht mit der komplizierten und zeitkritischen Programmierung des FDC herumschlagen (siehe auch unseren Beitrag auf Seite 38 dieser Ausgabe). Die vorhandenen ROM-Routinen sind so flexibel, daß durch entsprechende Änderung von Tabellen und Übergabeparametern jeder bislang beschriebene Schutz zu realisieren ist. Der Programmierer spart bei der Entwicklung des Schutzes viel Zeit und das Programm muß keine aufwendigen eigenen Laderoutinen enthalten, was dem ohnehin arg begrenzten Speicherplatz des Computers zugute kommt. Das bedeutet natürlich auch, daß Kopierer ein leichtes Spiel haben, da sie mit ihren Hilfsprogrammen ebenfalls auf die DOS-Routinen zugreifen können.

Aus diesem Grund häufen sich in letzter Zeit Programme, deren Lader direkt auf den FDC zugreifen. Kunstgriffe wie »gelöschte Data-Adress-Marks« finden sich immer öfter auf käuflichen Disketten. Wo soll das Ganze hinführen? Man könnte meinen, daß dies eine Entwicklungsspirale ohne Ende sei. Jedes noch so intelligente Kopierprogramm wird von einem noch besseren Schutz überlistet (oder umgekehrt). Doch ein Ende ist absehbar. Da der FDC ein recht intelligenter Bursche ist, haben die Entwickler des Chips viele komplexe Funktionen in einfache Befehle gepackt. Genau dort ist die Grenze gezogen, über die niemand hinauskommt. Sein begrenzter Befehlssatz schränkt auch seine Fähigkeiten ein. Beim CPC ist der Controller, im Gegensatz beispielsweise zum Commodore 64, nicht völlig frei programmierbar. Schon jetzt ist die Flexibilität des FDC soweit ausgeschöpft, daß neue Methoden zum Programmschutz nur noch schwer denkbar sind. Die Zukunft der Schützer sieht also alles andere als rosig aus, sie stehen auf verlorenem Posten. Vielleicht gehen deshalb immer mehr Firmen jetzt dazu über, ihre Produkte im wahrsten Sinne des Wortes »schutzlos auszuliefern« und auf die Ehrlichkeit des Großteils der Benutzer zu vertrauen.

Wir sollten ihr Vertrauen nicht mißbrauchen. (Oliver Suttorp/ja)

Ganz einfach: Grafik auf dem CPC

Die Grafikbefehle des Schneider CPC sind im Handbuch teilweise nur unzureichend und ohne Beispiele erklärt. Insbesondere das Handbuch des CPC 464 weist in dieser Hinsicht eklatante Schwächen auf. Wenn Sie die Grafik des CPC beherrschen und beispielsweise auch die relativen Grafikbefehle MOVER, PLOTR, DRAWR und TESTR optimal einsetzen möchten, ist dieser Beitrag genau richtig für Sie.

Um bei der Programmierung der Grafik des CPC überhaupt etwas zu erreichen, muß man erst einmal deren grundlegenden Aufbau kennen.

Der Bildschirm des CPC ist in allen drei Bildschirmmodi (Modus 0 bis 2) in ein feines Raster aus Grafikpunkten aufgeteilt. Im Modus 2 lassen sich 640 Punkte horizontal und 200 Punkte vertikal ansprechen, Modus 1 stellt 320 Punkte horizontal und 200 Punkte vertikal zur Verfügung, und Modus 0 bietet 160 horizontale und 200 vertikale Punkte.

Wie Sie sehen, ist die Zahl der vertikalen Punkte konstant, lediglich die Anzahl der horizontalen Punkte variiert und bestimmt damit die Auflösung der Grafik, das heißt, die Anzahl von Grafikpunkten pro Flächeneinheit. Je geringer die Auflösung der Grafik, desto mehr Farben lassen sich auf dem CPC darstellen. Der Grund liegt darin, weil durch eine geringere Auflösung weniger Grafikpunkte gespeichert werden müssen und dementsprechend mehr Farbinformationen pro Punkt im Bildspeicher abgelegt werden können.

Folgende Rechnung macht dies deutlich:

```
Auflösung:
Modus 2: 640 x 200 = 128000 Bildpunkte
Modus 1: 320 x 200 =  64000 Bildpunkte
Modus 0: 160 x 200 =  32000 Bildpunkte
Farben:
Modus 2:  2 Farben = 1 Bit Speicherbedarf
Modus 1:  4 Farben = 2 Bit Speicherbedarf
Modus 0: 16 Farben = 4 Bit Speicherbedarf
```

Rechnet man nun den Gesamt-Speicherbedarf für ein Bild in allen drei Modi aus, so erhält man jedesmal das gleiche Ergebnis:

```
Modus 2: 128000 x 1 Bit = 16000 Byte
Modus 1:  64000 x 2 Bit = 16000 Byte
Modus 0:  32000 x 4 Bit = 16000 Byte
```

Da der Bildspeicher des CPC 16 KByte (gleich 16384 Byte) groß ist, wird so der Speicherplatz in jedem Bildschirmmodus optimal ausgenutzt.

Die einzelnen Grafikpunkte des CPC lassen sich mit den Basic-Befehlen manipulieren, indem sie über ein rechtwinkliges Koordinatensystem angesprochen werden. Die Position jedes Punktes legen jeweils zwei Werte fest. Der erste Wert (x) gibt an, wie weit der Punkt parallel zur horizontalen Achse des Koordinatensystems vom Koordinaten-Nullpunkt (Achsen-Schnittpunkt) entfernt ist, der zweite Wert (y) gibt den Abstands des Punktes parallel zur vertikalen Achse vom Koordinaten-Nullpunkt an. Sicherlich ist Ihnen dieses Koordinatensystem noch aus der Schulzeit bekannt (Bild 1).

Die Grafikbefehle des CPC arbeiten immer mit einem Koordinatensystem, das 640 Punkte in der horizontalen und 400 Punkte in der vertikalen Richtung erlaubt, ganz gleich, welche Auflösung der gewählte Bildschirm-Modus tatsächlich zuläßt. So adressieren beispielsweise im Modus 1 die Koordinaten 100/100, 101/100, 100/101 und 101/101 alle den gleichen Punkt auf dem Bildschirm.

Das Verhältnis von 640 horizontalen zu 400 vertikalen Punkten wurde deshalb gewählt, weil auf diese Weise ein Bildpunkt (zwei übereinanderliegende Punkte im Modus 2) annähernd quadratisch ist, und beispielsweise ein Kreis auch wirklich wie ein Kreis und nicht wie ein Oval aussieht (das optimale Verhältnis von horizontalen zu vertikalen Punkten beträgt 640 zu 480).

Damit das Grafik-Koordinatensystem des CPC dem mathematischen Koordinatensystem entspricht, befindet sich der Nullpunkt nach dem Einschalten links unten auf dem Bildschirm und die Koordinaten-Positionen zählen von links nach rechts beziehungsweise von unten nach oben.

Wenn wir im folgenden die Grafikbefehle des Schneider CPC besprechen, teilen wir sie grob in absolute und relative Befehle auf. Bei absoluten Grafikbefehlen werden die Koordinaten eines Bildpunktes immer ausgehend vom Nullpunkt berechnet, bei relativen werden die im Befehl angegebenen Koordinaten ausgehend von der aktuellen Position des Grafikcursors verwendet. Das heißt die momentane Position des Grafikcursors wird als Koordinaten-Nullpunkt betrachtet.

Einer der wichtigsten Grafikbefehle des CPC ist der PLOT-Befehl. Der PLOT-Befehl setzt einzelne Grafikpunkte auf dem Bildschirm. Das Befehlsformat lautet

```
PLOT xordinate, yordinate, farbregister
```

Für Farbregister ist 1 der Standardwert. Wünschen Sie keine Änderung des Farbregisters, brauchen Sie diesen Wert nicht anzugeben. Der Computer nimmt in diesem Fall immer die Farbe des zuletzt gewählten Farbregisters an.

Punkt für Punkt mit PLOT

Mit dem PLOT-Befehl und den trigonometrischen Funktionen COS und SIN läßt sich auf einfache Weise ein Kreis zeichnen:

```
10 DEG:MODE 2
20 ORIGIN 320,200
30 FOR a=1 TO 360
40 PLOT 100*COS(a),100*SIN(a)
50 NEXT a
```

In diesem kleinen Listing kommt bereits der ORIGIN-Befehl vor. ORIGIN verschiebt den Nullpunkt des Koordinatensystems an eine beliebige Position des Bildschirms, in unserem Fall auf die Bildschirmmitte.

Die Variable <a> enthält den aktuellen Winkel für die Sinus- und Kosinusfunktion. Um einen kompletten Kreis zu zeichnen, wird eine volle Umdrehung von 360 Grad benötigt. Der Sinus von a ist dabei nichts anderes als die Y-Ordinate eines Punktes, der Teil eines Kreises mit dem Radius 1 ist, und der Kosinus von a entspricht der X-Ordinate eines Punktes dieses Kreises (Bild 2).

Soll der Kreis einen größeren Radius erhalten, so müssen Sie den Sinus- und Kosinuswert mit dem gewünschten Radius multiplizieren. Sie dürfen jedoch diesen Wert nicht zu hoch wählen, weil Ihnen nach oben

und unten nur jeweils 200 Bildpunkte zur Verfügung stehen.

Multiplizieren Sie beispielsweise den Sinus- und Kosinuswert mit unterschiedlichen Radien, so erhalten Sie eine Ellipse. Ändern Sie das Listing in Zeile 40 wie folgt:

```
40 PLOT 50*cos(a),150*sin(a)
```

Sie können mit den beiden Multiplikatoren getrost experimentieren oder beispielsweise Zufallsellipsen über den Bildschirm verteilen. Versetzen Sie den ORIGIN-Befehl über RND-Werte und lassen Sie jeweils eine Ellipse mit Zufalls-Multiplikatoren zeichnen.

Bewegung kommt ins Spiel

Der Befehl MOVE ist mit dem PLOT-Befehl eng verwandt und funktioniert fast genauso. Mit dem MOVE-Befehl läßt sich der Graphikcursor auf die angegebene Koordinate setzen, jedoch wird kein sichtbarer Punkt an dieser Position erzeugt. Das Befehlsformat des MOVE-Befehls lautet

```
MOVE xordinate, yordinate, farbregister
```

Der Befehl DRAW zieht eine Linie von der aktuellen Position des Grafikcursors bis zur angegebenen Koordinate. Das Befehlsformat lautet:

```
DRAW xordinate, yordinate, farbregister
```

Betrachten Sie noch einmal das Listing, das einen Kreis zeichnet. Ändern Sie die Zeile 40 und setzen Sie mit dem MOVE-Befehl den Grafikcursor in der FOR-NEXT-Schleife immer wieder auf Null zurück, dann entsteht ein sternförmiges Gebilde. Fügen Sie dazu Zeile 35 in das Programm neu ein und ändern Sie Zeile 40 folgendermaßen ab:

```
35 MOVE 0,0
40 DRAW 100*COS(a),100*SIN(a)
```

Um die Anzahl der Strahlen zu bestimmen, bauen Sie in die FOR-Zeile noch einen STEP-Befehl (Schrittweite) ein. Dabei gilt:

```
Schrittweite = 360 / Strahlenanzahl
```

Wünschen Sie beispielsweise 20 Strahlen, so ändern Sie Zeile 30 zu

```
30 FOR a=1 to 360 STEP 18
```

Nachdem Sie jetzt die drei wichtigsten Grafikbefehle des CPC kennen, ist es für Sie ein leichtes, ein beliebiges Rechteck zu zeichnen. Probieren Sie einmal, ein Quadrat mit der Seitenlänge von 100 auf die Mitte des Bildschirms ausgeben zu lassen. Dazu gibt es mehrere Wege. Wer nicht lange zögert und sofort losprogrammiert, dessen Listing wird folgendermaßen aussehen:

```
10 MOVE 270,150
20 DRAW 370,150
30 DRAW 370,250
40 DRAW 270,250
50 DRAW 270,150
```

Doch wer sich ein paar Gedanken macht, kann sich mit dem ORIGIN-Befehl Umrechnungsarbeit ersparen.

```
10 ORIGIN 270,150
20 DRAW 100,0
30 DRAW 100,100
40 DRAW 0,100
50 DRAW 0,0
```

Und der Perfektionist, der gerne im Mittelpunkt steht, schreibt:

```
10 ORIGIN 320,200
20 MOVE -50,-50
30 DRAW 50,-50
40 DRAW 50,50
50 DRAW -50,50
60 DRAW -50,-50
```

Wenn Ihnen nicht ganz klar ist, wie diese Beispiele funktionieren, versuchen Sie am besten mit Papier und Bleistift die Bewegungen des Grafikcursors nachzuvollziehen.

Der Zweck von DEG

Bis jetzt wurde die Ausgabe des Befehls DEG in der ersten Zeile des Kreis-Programms noch nicht erläutert. Der DEG-Befehl ist nötig, weil das Programm bei den Befehlen COS und SIN im Gradmaß arbeiten soll und DEG dieses Gradmaß einschaltet. (Der CPC ist nach dem Einschalten auf das Bogenmaß eingestellt, das durch den Befehl RAD auch wieder eingeschaltet werden kann).

Daß der ORIGIN-Befehl den Koordinaten-Nullpunkt festlegt, wurde bereits erwähnt, doch man kann mit diesem Befehl auch ein Grafikfenster bestimmen, dessen linke untere Ecke dann dem Koordinaten-Nullpunkt gleichgesetzt wird. Das normale Befehlsformat von ORIGIN lautet

```
ORIGIN xordinate, yordinate
```

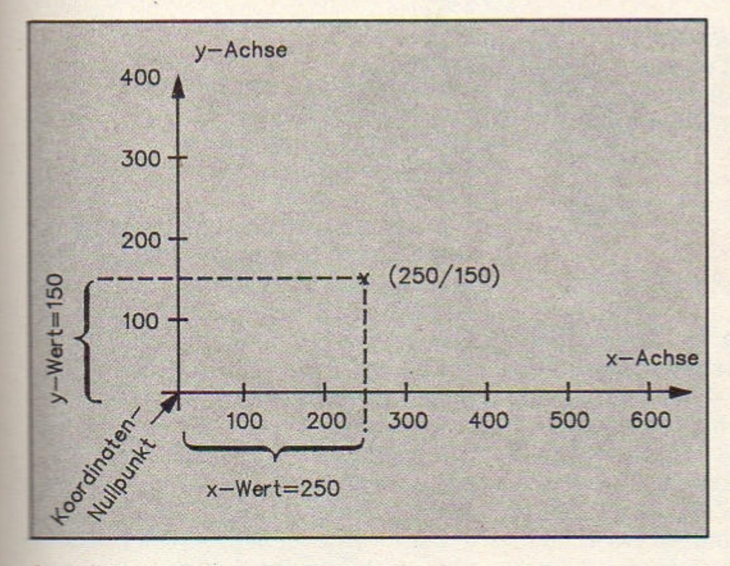
Kommt dann noch ein Grafikfenster hinzu, so hat der Befehl folgendes Format:

```
ORIGIN xordinate, yordinate, links,
rechts, oben, unten
```

Ein Beispiel:

```
MODE 1:ORIGIN 100,100,100,540,300,100
```

Um dieses Fenster auch sichtbar zu machen, müssen Sie beispielsweise »CLG 3« eingeben. Das Grafikfenster wird nun mit der Farbe aus Farbregister 3 gefüllt. Der Befehl CLG arbeitet ähnlich wie der CLS-Befehl, nur löscht er nicht den gesamten Bildschirm, sondern nur das Grafikfenster in der Farbe des angegebenen Farbregisters.



▲ **Bild 1. Nach diesem Prinzip lassen sich Punkte mit zwei Angaben in einem rechtwinkligen Koordinatensystem adressieren**

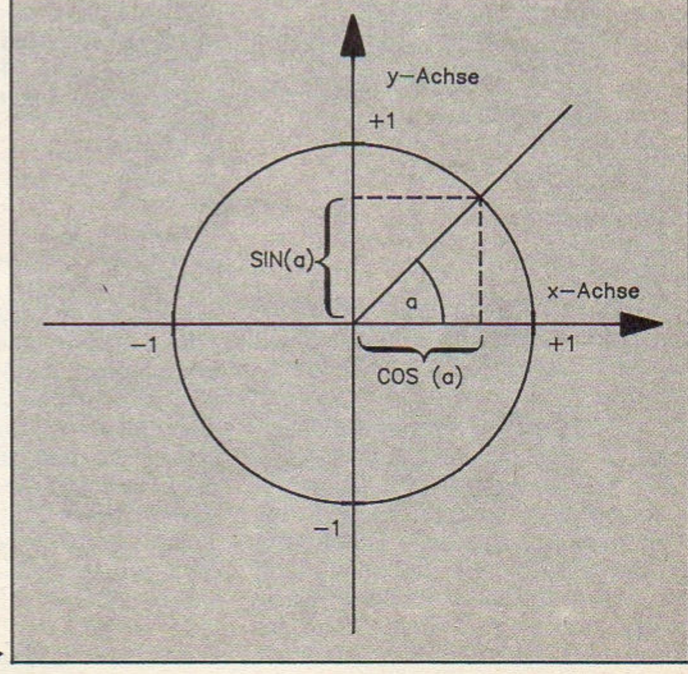


Bild 2. So läßt sich mit den Funktionen COS und SIN ein Kreis zeichnen ►

Selten benötigt man bei der Grafikprogrammierung den Befehl TEST. Mit TEST können Sie überprüfen, welches Farbregister auf der angegebenen Koordinate verwendet wurde. Das Befehlsformat lautet

```
TEST (xordinate,yordinate)
```

Folgendes kleine Beispiel verdeutlicht die Funktion des TEST-Befehls:

```
PLOT 100,100,3
PRINT TEST(100,100)
```

Nachdem wir alle absoluten Grafikbefehle des CPC behandelt haben, wenden wir uns nun den relativen Grafikbefehlen zu.

Der Befehl PLOTR setzt wie PLOT einen Punkt, nur daß der Computer die Koordinaten relativ zur aktuellen Position des Grafikcursors betrachtet. Befehlsformat:

```
PLOTR xversatz, yversatz, farbregister
```

Das folgende Beispiel setzt die Ecken einer Raute auf den Bildschirm:

```
10 PLOT 100,100
20 PLOTR 10,10
30 PLOTR -10,10
40 PLOTR -10,-10
```

Auch der Befehl DRAWR arbeitet relativ. Er eignet sich hervorragend für dreidimensionale Darstellungen. Das Befehlsformat lautet

```
DRAWR xversatz, yversatz, farbregister
```

Folgendes Programm zeichnet einen dreidimensionalen Würfel:

```
10 MODE 1
11 'Bildschirm-Initialisierung
20 c=1
21 'c=Farbregister
30 ORIGIN 320,200
31 'Definition des Nullpunktes
40 FOR x=40 TO 0 STEP -2
41 'Schleife zum Aufbau des Wuerfels
50 MOVE x,x
51 'linke, untere Quadratecke
60 IF x(40 THEN c=3:'erster Durchlauf?
70 GOSUB 110
80 NEXT
90 c=1:GOSUB 110
91 'letztes Quadrat zeichnen
100 END
110 'UP Quadrat zeichnen
120 DRAWR 100,0,c
130 DRAWR 0,100
140 DRAWR -100,0
150 DRAWR 0,-100
160 PLOT x,x,1
170 RETURN
```

Die REM-Zeilen müssen Sie selbstverständlich nicht abtippen; sie gelten lediglich der Programm-Dokumentation.

Das Bild eines Würfels entsteht dadurch, daß die linke untere Ecke eines Quadrates auf den Nullpunkt zubewegt wird. In Zeile 60 prüft das Programm, ob der erste Durchgang der Schleife vorliegt, und somit noch ein helles Quadrat für den Rand gesetzt werden muß. Darauf folgt der Aufruf des Unterprogramms »Quadrat zeichnen« und das Quadrat wird mit der Farbe aus dem aktuellen Farbregister gezeichnet. Am Ende der Routine wird in die linke untere Ecke ein heller Punkt gesetzt, um den dreidimensionalen Effekt zu verstärken. Wenn die Programmschleife abgearbeitet ist, wird zum Schluß noch ein helles Quadrat davorgesetzt.

Der relative Grafikbefehl MOVER versetzt den Grafikcursor von seiner aktuellen Position um <xversatz> Bildpunkte horizontal und um <yversatz> Bildpunkte vertikal. Das Befehlsformat lautet:

```
MOVER xversatz, yversatz, farbregister
```

Ein weiterer relativer Befehl ist die TESTR-Funktion, die das Farbregister anzeigt, das auf dem angegebenen Versatz zu der aktuellen Grafikcursorposition verwendet wurde. Befehlsformat:

```
PRINT TESTR(xversatz,yversatz)
```

Neben den absoluten und relativen Grafikbefehlen gibt es noch einige weitere Befehle, die bei der Grafikprogrammierung eine wichtige Rolle spielen. Einer dieser Befehle ist TAG. Der TAG-Befehl erlaubt das Mischen von Text und Grafik. Sie werden jetzt vielleicht einwenden, daß sich dies ohnehin schon mit dem LOCATE- und PRINT-Befehl machen läßt. Doch im TAG-Modus funktioniert das Schreiben etwas anders: Die Zeichen werden nicht mehr auf die Position des Textcursors, sondern auf die Position des Grafikcursors ausgegeben, so daß sich Text nun auch zwischen den Textzeilen darstellen läßt.

Der Befehl TAGOFF schaltet den TAG-Modus wieder aus. Die Formate für die beiden Befehle lauten:

```
TAG #fenster
TAGOFF #fenster
```

(Wenn das Standardfenster 0 angesprochen ist, reicht es aus, nur »TAG« oder »TAGOFF« zu schreiben.)

Ein Beispiel zum TAG-Befehl:

```
10 TAG
20 MOVE 100,310
30 PRINT"Dieser Text steht zwischen
den Zeilen.";
40 MOVER 16,-24
50 PRINT"Wie sieht das aus ?";
60 TAGOFF
```

Wie im normalen Grafikmodus erfolgt auch in diesem Fall keine Fehlermeldung, wenn eine Textausgabe über den Bildrand hinweg erfolgt.

Wichtig beim Arbeiten mit dem TAG-Befehl ist, daß sämliche Steuerzeichen (das sind die Zeichen mit ASCII-Werten unter 32) mit einem PRINT-Befehl nicht mehr ausgeführt, sondern auf dem Bildschirm angezeigt werden. So wird zum Beispiel nach jedem PRINT der Cursor nicht auf die erste Stelle der nächsten Zeile gesetzt, weil dazu die Steuerzeichen 10 und 13 verwendet werden müßten. Auch Befehle wie CLS, PEN oder PAPER ignoriert der Computer.

Die beiden Befehle XPOS und YPOS funktionieren wie Variablen, die ständig die aktuelle (und absolute) X- beziehungsweise Y-Ordinate der Position des Grafikcursors enthalten. Ein Beispiel:

```
10 MOVE 225,350
20 PRINT"X-Position :";XPOS,"Y-
Position:";YPOS
```

Bislang war, wenn es um Farben ging, immer nur von Farbregistern, doch nie von den Farbwerten selbst die Rede. Der Befehl INK ist dazu gedacht, einem Farbregister einen bestimmten Farbwert zuzuweisen. Es können die Farbregister 0 bis 15 mit den Farbwerten 0 bis 26 belegt werden. Das Befehlsformat lautet:

```
INK farbregister, farbwert
```

Wird ein zweiter Farbwert angegeben, so blinken diese beiden Farben abwechselnd (die Geschwindigkeit des Blinkens läßt sich mit dem Befehl SPEED INK einstellen).

Im nachfolgenden Programm wird durch das Wechseln der Farbwerte in den Farbregistern die Bewegung einer Kugel simuliert:

```
10 MODE 1:DEG
20 DEFINT a-z
30 ORIGIN 320,200
40 FOR i=1 TO 3:INK i,26:NEXT:INK 0,0
50 m=100:n=100:c=1:d=3
60 MOVE 0,n
70 FOR a=1 TO 360 step 5
80 IF a(180 THEN PLOT m*SIN(a),n*COS
(a),c ELSE PLOT m*SIN(a),n*COS(a),d
90 NEXT:c=c+1:IF c)3 THEN c=1
100 d=d-1:IF d(1 THEN d=3
110 m=m-5
120 IF m)=0 THEN 60
130 f=10
140 FOR i=1 TO 3
150 INK i,f
160 f=f+5:IF f)20 THEN f=5
170 FOR w=1 TO 50:NEXT
180 NEXT
190 GOTO 140
```

Die Kugel wird erzeugt, indem mehrere Ellipsen übereinander gelegt und jeweils nach einer halben Ellipse die Farbwerte der Farbregister gewechselt werden. Ohne Farbwechsel wäre auf beiden Seiten der Kugel eine Drehung von außen nach innen zu sehen.

Nun sind wir am Ende unseres Streifzuges durch die Welt der Grafikprogrammierung angelangt. In der Basic-Version 1.1 des CPC 664 und 6128 gibt es zwar noch einige zusätzliche Grafikbefehle, diese werden jedoch im zugehörigen Handbuch ausführlich erklärt, so daß sich eine Besprechung an dieser Stelle erübrigt. (Christian Aschoff/ma)

Was Sie schon immer über GSX wissen wollten

GSX ist eine Grafikerweiterung für CP/M Plus, die im Lieferumfang des CPC 6128 bereits enthalten ist. Da jedoch jegliche Dokumentation zu dieser Software fehlt, ist die Art der Anwendung unter den 6128-Besitzern relativ unbekannt. Der folgende Beitrag erklärt Ihnen, was es mit GSX auf sich hat.

Durch den rasanten Anstieg von Leistung und Geschwindigkeit auf dem Gebiet der Computerentwicklung sowie durch den anhaltenden Preisverfall bei elektronischen Bausteinen werden auch die Grafikfähigkeiten der Computer immer besser und preiswerter.

So ist Computergrafik inzwischen ein Thema, das viele Computer- und Nicht-Computerbesitzer mit großem Interesse verfolgen. Neue Einsatzgebiete in Industrie und Wissenschaft, in der Werbung und nicht zuletzt im Heimbereich faszinieren und begeistern auch technisch weniger interessierte Menschen.

Viele neue Programmiertechniken wie zum Beispiel der Einsatz von CAD (computerunterstützte Konstruktion) und CAM (computerunterstützte Produktion) wurden durch hohe Grafikauflösung und eine breite Farbenpalette erst möglich.

Die aktuellen Betriebssysteme oder zumindest Teile davon, wie der Finder des Macintosh oder GEM auf dem Atari ST und Windows auf dem PC/AT arbeiten grafikorientiert, so daß auch der Bedienungskomfort für den Benutzer deutlich gestiegen ist.

Aufgrund dieser Entwicklung ergeben sich natürlich Probleme mit älteren Betriebssystemen. Da diese hauptsächlich für die Ein- und Ausgabe von Texten konzipiert wurden, fehlten lange Zeit die Grafikanwendungen. Dies liegt nicht zuletzt auch darin begründet, daß die Entwickler der Betriebssysteme Informationen über die Arbeitsweise ihrer Routinen lange Zeit zurückhielten.

Dies galt lange Zeit auch für das 8-Bit-Betriebssystem CP/M, das neben MS-DOS und Unix zu den am weitesten verbreiteten Betriebssystemen zählt.

Doch Digital Research, der Hersteller von CP/M, hatte ein Einsehen und entwickelte zu der verbesserten Betriebssystem-Version CP/M Plus die universelle Grafikerweiterung GSX (graphics system extension).

Leider wurde GSX in seiner Leistungsfähigkeit bislang weit unterschätzt und viel zu wenig beachtet, doch das kann sich ändern.

Beim Kauf des Schneider CPC 6128 wird nämlich neben den Betriebssystemen CP/M 2.2 und CP/M Plus sowie der Programmiersprache Logo auch die Grafikerweiterung GSX mitgeliefert, die dem CP/M-Plus-Benutzer Zugang zur Programmierung von Grafik unter diesem Betriebssystem verschafft.

GSX – ein Geheimnis?

Aus unerfindlichen Gründen bleibt GSX jedoch in der begleitenden Dokumentation des CPC 6128 unerwähnt, von einer näheren Beschreibung ganz zu schweigen. Es finden sich allenfalls ein paar Anmerkungen, die auf die Leistungsfähigkeit von GSX hinweisen und die Anpassung von bereits geschriebenen, professionellen Programmen an die Grafikroutinen erläutern.

Dieses Verhalten der Firma Schneider ist insbesondere merkwürdig, da sich mit der GSX-Erweiterung recht schnell saubere Zeichnungen anfertigen lassen, wie die simple Grafik in Bild 1 beweist.

GSX ist allgemein betrachtet ein Programm, das bestimmte Datenstrukturen und Anwendungsgebiete zusammenfaßt, so daß ein vollkommen Hardware-unabhängiges System entsteht, das inzwischen auf vielen verschiedenen Computern eingesetzt wird. Selbst unter MS-DOS und CP/M 80 gibt es Implementierungen.

Bild 2 zeigt, auf welchen Systemen GSX mittlerweile eingesetzt wird.

Zu Beginn möchten wir Ihnen am Beispiel von Turbo-Pascal demonstrieren, wie die Anpassung der Grafikerweiterung GSX an das Betriebssystem funktioniert.

Grundsätzlich bestehen zwei Möglichkeiten, Turbo-Pascal-Programme mit dem GSX-Zusatz zu versehen.

Zum einen können Sie sämtliche der mit Hilfe der Compiler-Option »COM« erzeugten Turbo-Pascal-Programme einzeln GSX-fähig machen. Diese Alternative hat aber den Nachteil, daß Sie die auf diese Weise modifizierten Programme jedesmal neu anpassen müssen, wenn Sie Änderungen im Quellcode vornehmen.

Zum anderen können Sie die Betriebssystemumgebung von Turbo-Pascal selbst verändern, so daß alle compilierten Programme sofort ausprobiert werden können. Im folgenden werden wir diesen Weg für Turbo-Pascal-Programme realisieren.

Zu diesem Zweck müssen Sie zuerst einige Dateien auf die Turbo-Pascal-Programmdiskette kopieren, die später genauer erläutert werden.

Auf den Systemdisketten des CPC 6128 befinden sich drei Dateien, die Sie auf jeden Fall kopieren müssen. Diese sind »gsx.sys«, »assign.sys« und »gengraf.com«.

Des weiteren benötigen Sie mindestens einen Treiber, auf dessen Bedeutung wir später noch eingehen werden. Sollten Sie die genannten Dateien auf Ihrer Turbo-Pascal-Diskette installiert haben, so geben Sie bitte folgende Befehlszeile unter CP/M Plus ein:

```
gengraf turbo
```

Nach einiger Zeit erscheint auf dem Bildschirm eine Meldung, die Ihnen mitteilt, daß GSX installiert ist.

Die Installation von GSX

Jetzt können Sie Turbo-Pascal so starten, so wie Sie es bislang gewohnt waren. Doch werden Sie überrascht feststellen, daß Ihnen wesentlich weniger Speicherbereich für Ihre Programme zur Verfügung steht als zuvor. Der Speicherplatz ist deshalb eingeschränkt, weil sich nun im Arbeitsspeicher des CPC neben Turbo-Pascal auch die GSX-Erweiterung befindet. Dadurch fehlen unter der Turbo-Pascal-Version 3.0 immerhin fast zwölf KByte an freiem Speicherplatz.

Wenden wir uns nun den grundlegenden Eigenschaften von GSX zu. GSX erweitert die Eigenschaften von CP/M Plus generell um ein grafisches Ausgabegerät. In Wirklichkeit existiert dieses Ausgabegerät jedoch nicht. Es wird nur der Programmiersprache

beziehungsweise dem angepaßten Programm als weitere Ausgabeeinheit angeboten. Die eigentliche Anpassung an die tatsächlich angeschlossene Peripherie wird dann vom GSX-System und den sogenannten Treibern durchgeführt.

So ist GSX für jedes Peripheriegerät geeignet. Es müssen nur die entsprechenden Treiber zur Verfügung stehen. Diese müssen zusätzlich zu den Installationsprogrammen und dem GSX-System auf die Arbeitsdiskette kopiert werden.

Auf den Systemdisketten des CPC 6128 befinden sich folgende Treiber:

```
ddmode0.prl
ddmode1.prl
ddmode2.prl
dd-dmp1.prl
ddfxlr7.prl
ddshinwa.prl
ddhp7470.prl
```

Die Dateien »ddmode0«, »ddmode1« und »ddmode2« sind die Treiber für die drei Bildschirmmodi des CPC (MODE 0, MODE 1 und MODE 2).

Der Treiber »dd-dmp1« eignet sich für Ausgaben auf den Schneider-Drucker DMP 2000 und »ddfxlr7« ist für den Ausdruck auf Epson-Druckern beziehungsweise Epson-kompatiblen Druckern vorgesehen.

Die Datei »ddshinwa« ist für spezielle Peripherie vorhanden, die den Shinwa-Mechanismus benutzt, und »ddhp7470« bedient den Hewlett-Packard-Plotter 7470 und kompatible Geräte.

Die Namen der im Zusammenhang mit GSX verwendeten Treiber müssen sich in der ASCII-Datei »assign.sys« befinden. In dieser Datei sind auch die Gerätenummern gespeichert, durch deren Aufruf ein bestimmter Teil der Peripherie angesprochen wird. Dabei ist es wichtig, daß die Treiber der Größe nach, mit dem längsten beginnend, aufgeführt sind.

GSX lädt nämlich beim ersten Aufruf den ersten Treiber in den Programmspeicher und reserviert den Speicherplatz für diese Datei. Sollte später ein anderer Treiber benötigt werden, so wird dieser in diesen reservierten Arbeitsbereich geladen. Aus diesem Grund darf der später benötigte Platz nicht größer sein als der zuvor definierte. Andernfalls könnte das Betriebssystem abstürzen.

Die Gerätenummern werden in der Regel nach folgendem Prinzip benutzt:

01 bis 10: Bildschirm
11 bis 20: Drucker
21 bis 30: Plotter

Als Demonstrationsbeispiel für die Anwendungen von GSX dient uns ein Turbo-Pascal-Programm, dessen einfache Aufgabe es ist, auf dem Bildschirm unter CP/M Plus einen kurzen Text anzuzeigen und ein Rechteck zu zeichnen.

GSX ist aber nicht auf Pascal festgelegt, sondern arbeitet grundsätzlich mit jeder Programmiersprache zusammen beziehungsweise mit jeder Datei mit der Extension »com«, die es erlaubt, Systemaufrufe zu programmieren.

GSX arbeitet wie die Basic-Grafik des CPC mit einem zweidimensionalen Koordinatensystem, dessen Nullpunkt sich links unten auf dem Bildschirm befindet. Die beiden Koordinatenwerte x und y können jeweils Werte von 0 bis 32767 annehmen.

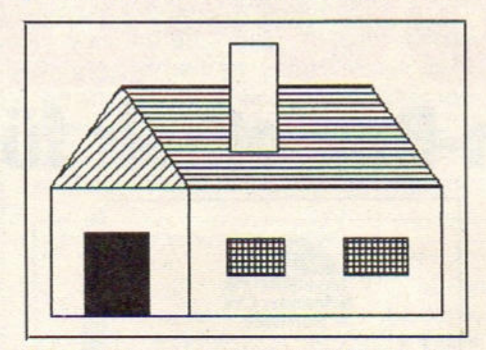


Bild 1. Fix gezeichnet: Ein Häuschen unter CP/M und GSX

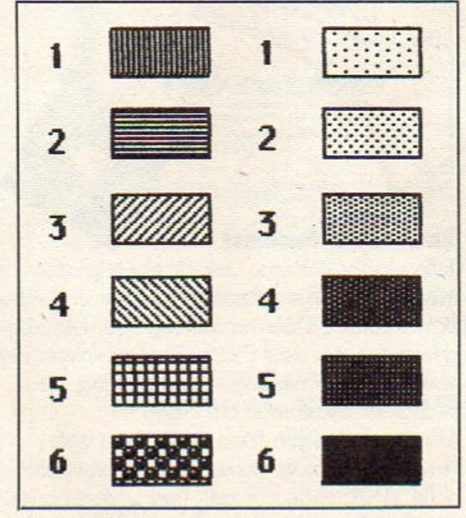


Bild 3. Sechs verschiedene Füllmuster und sechs verschiedene Schraffuren beherrscht GSX

Einige Fähigkeiten von GSX werden allerdings nicht durch alle Treiber beziehungsweise von allen angeschlossenen Peripheriegeräten unterstützt. So ist es zum Beispiel bei einem Plotter grundsätzlich möglich, Schriften in verschiedenen Winkeln auszugeben; bei einem Drucker oder Monitor versagt dieses Verfahren jedoch.

GSX stellt dem angesprochenen Programm eine Art Betriebssystemumgebung namens GDOS (graphics device operating system) zur Verfügung, die wie CP/M bestimmte Systemroutinen anbietet. Ein GSX-Grafikbefehl wird dabei ähnlich aufgerufen wie eine Betriebssystemroutine.

Die Befehle von GSX können Kreise oder Kreisausschnitte zeichnen, verschiedene Farben auswählen, unterschiedliche Linien ziehen, Text in mehreren Formaten schreiben, gefüllte Flächen malen und vieles mehr.

GSX praktisch eingesetzt

Besonders hervorzuheben sind die verschiedenen Darstellungsarten an gefüllten Flächen, die in Bild 3 aufgelistet sind. So kann man unter GSX zum Beispiel mit einem bestimmten Muster oder einer gewünschten Schraffur ausgefüllte Rechtecke, Kreise oder Polygone zeichnen lassen.

Gesteuert werden die GSX-Kommandos unter CP/M mit Hilfe des BDOS-Aufrufs 115, wobei das Register DE die Adresse eines Parameterblocks angeben muß, der die einzelnen Parameter der gewählten Funktion enthält.

Dies hört sich jetzt sehr kompliziert an, doch das abgedruckte Turbo-Pas-

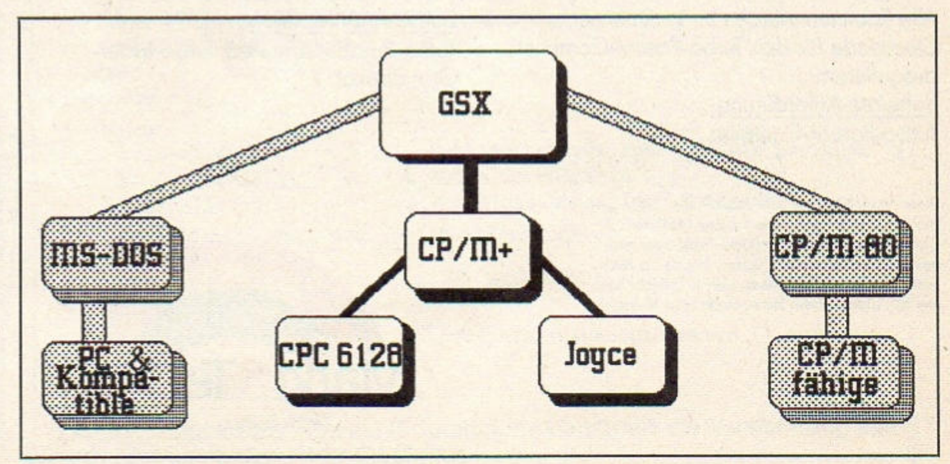


Bild 2. GSX wird nicht nur auf dem Computermodell CPC 6128 eingesetzt

Farben	Code
Schwarz	0
Rot	1
Grün	2
Blau	3
Zyan	4
Gelb	5
Magentarot	6
Weiß	7
Kreise	**Code**
Kreisbogen	2
Zylinder	3
Linien	**Code**
durchgehend	1
gestrichelt	2
gepunktet	3
Strich-Punkt	4
lange Striche	5
Strich-Punkt-Punkt	6
Markierungen	**Code**
Punkt	1 (.)
Plus	2 (+)
Stern	3 (*)
Kreis	4 (o)
Kreuz	5 (x)
Schraffuren	**Code**
Senkrecht	1
Waagrecht	2
+ 45 Grad	3
+ 135 Grad	4
Gitter	5
Gitter (diagonal)	6
Füllart	**Code**
Rahmen	1
Durchgehend	2 (siehe unter Linien)
Halbton	3
Schraffiert	4
Darstellung	**Code**
Invertiert	13
Normal	14
Cursor	**Code**
nach oben	4
nach unten	5
nach rechts	6
nach links	7
nach 0,0	8 (Cursor an Bildschirm-anfang)
bis Bildschirmende löschen	9
bis Zeilenende löschen	10
Zeichensätze	**Code**
verschieden, je nach Gerätetyp – jedoch Standard	1
Zeichenmodi	**Code**
Überschreiben	1
Transparent	2
Ergänzen	3
Löschen	4
Arbeitsgeräte	**Code**
Keyboard	1
Maus/Joystick	2
Eingabemodi	**Code**
auf Bestätigung der Eingabe warten	1
Eingabe lesen und sofort weiter	2

Wertebereiche einzelner GSX-Funktionen

cal-Listing beinhaltet eine Prozedur namens GDOS, die das erklärte Prinzip verdeutlicht. Sie übergeben der Prozedur einfach eine Liste von Parametern, die sowohl die Nummer der gewünschten Funktion, als auch Angaben über die Anzahl der Koordinaten und schließlich die Koordinaten selbst enthält.

Die Tabelle zeigt eine Aufstellung der Wertebereiche einzelner Funktionen, die Ihnen die Grafikprogrammierung sehr erleichtert. Die Funktionsnummern der GSX-Grafikbefehle können Sie dem Listing und der weiterführenden Literatur entnehmen.

Wir hoffen, daß dieser kleine Schnupperbeitrag zum Thema GSX Sie angeregt hat, sich etwas näher mit dieser Grafikerweiterung zu beschäftigen. Daraus erschließen sich Ihnen weitere interessante Anwendungsgebiete und Sie werden sehen, zu welchen erstaunlichen Leistungen GSX fähig ist. Größere Kosten entstehen dabei nicht, weil GSX dem CPC 6128 gratis beiliegt.

Damit Sie auch Gelegenheit haben, sich mit GSX näher zu befassen, empfehlen wir Ihnen den Kauf des GSX-Handbuchs von Digital Research, das der Verlag Markt&Technik für 39 Mark anbietet. Dieses Buch führt den Leser detailliert in die Grafikprogrammierung unter GSX ein. Es ist jedoch in englischer Sprache geschrieben.

Praktische Anwendungen für GSX und Routinen zum Einbinden in höhere Programmiersprachen finden Sie in verschiedenen Büchern über die Grafikprogrammierung des CPC sowie in den Fachbeiträgen und Kursen diverser Computerzeitschriften.

(Markus Zietlow/ma)

```
program gsxdemonstration;
 const
   monitor            = 1 ; (* Ausgabegeraet *)
   linienart          = 1 ; (* Nicht unterbrochen *)
   linienfarbe        = 7 ; (* Weiss *)
   grafikcursorform   = 2 ; (* Kreuz *)
   grafikcursorfarbe  = 7 ; (* Weiss *)
   zeichensatz        = 1 ; (* Systemzeichensatz *)
   textfarbe          = 7 ; (* Weiss *)
   fuellart           = 3 ; (* Schraffur *)
   fuellmuster        = 5 ; (* Gitter *)
   fuellfarbe         = 7 ; (* Weiss *)
 type
   feld_1 = array [1..10]  of integer;
   feld_2 = array [1..200] of integer;
   feld_3 = array [1..255] of integer;
   zeichenkette = string [255];
 var
   kontrollfeld                                   : feld_1;
   koordinatenpaare, koordinatenausgabeparameter : feld_2;
   ascii_werte     , ascii_ausgabeparameter      : feld_3;
   block : array [1..5] of integer;
   text  : zeichenkette;
   zaehler, linksx, linksy, rechtsx, rechtsy, x, y : integer;
procedure gdos (var kontrollfeld                                   : feld_1;
                var ascii_werte     ,ascii_ausgabeparameter      : feld_3;
                var koordinatenpaare,koordinatenausgabeparameter : feld_2 );
   begin
     block[1]:=addr(kontrollfeld);
     block[2]:=addr(ascii_werte);
     block[3]:=addr(koordinatenpaare);
     block[4]:=addr(ascii_ausgabeparameter);
     block[5]:=addr(koordinatenausgabeparameter);
     bdos(115,addr(block));
   end;
procedure clg;
   begin
     kontrollfeld[1] := 5 ; (* Funktionsaufruf fuer "Grafik loeschen" *)
     kontrollfeld[2] := 0 ; (* Keine Koordinaten *);
     kontrollfeld[6] := 2 ; (* Funktionsabhaengiger Parameter *)
     (* Initialisieren des Grafikbildschirms *)

     gdos (kontrollfeld     , ascii_werte                , ascii_ausgabe-
 parameter,
           koordinatenpaare , koordinatenausgabeparameter )
   end;
procedure locate (x,y:integer);
```

Listing. Dieses Turbo-Pascal-Programm demonstriert die Verwendung von GSX-Grafikbefehlen unter CP/M Plus

```
    begin
      (* Initialisieren der Positionsangabe *)
      kontrollfeld[1] := 5 ; (* Funktionsnummer *)
      kontrollfeld[2] := 0 ; (* Keine Koordinaten *)
      kontrollfeld[6] := 11; (* Position setzen*)
      (* Abspeichern der Koordinaten *)

      ascii_werte[1] := y;
      ascii_werte[2] := x;

      (* Cursor setzen *)

      gdos (kontrollfeld     , ascii_werte                 , ascii_ausgabepara-
  meter ,
            koordinatenpaare , koordinatenausgabeparameter)
    end;
   procedure print (text : zeichenkette);
    begin
      (* Umwandeln des Textes in das Parameterfeld *)
      for zaehler := 1 to length(text) do
        ascii_werte[zaehler] := ord(text[zaehler]);
      (* Festlegung der Funktionsparameter *)
      kontrollfeld[1] := 5            ; (* Funktionsnummer fuer Textausgabe *)
      kontrollfeld[2] := 0            ; (* Keine Koordinaten *)
      kontrollfeld[4] := length(text) ; (* Laenge der Zeichenkette *)
      kontrollfeld[6] := 12           ; (* Funktionsspezifischer Parameter *)
      (* Text an der aktuellen Cursorposition ausgeben *)
      gdos (kontrollfeld     , ascii_werte                , ascii_ausgabeparameter,
            koordinatenpaare , koordinatenausgabeparameter )
      end;
   procedure kasten (linksx ,linksy, rechtsx, rechtsy :integer);
    begin
      kontrollfeld[1] := 11      ; (* Funktionsnummer *)
      kontrollfeld[2] := 2       ; (* Anzahl der Koordinaten *)
      kontrollfeld[6] := 1       ; (* Funktionsabhaengiger Parameter *)
      (* Koordinaten der linken, unteren Ecke *)
      koordinatenpaare[1] := linksx ;
      koordinatenpaare[2] := linksy ;
      (* Koordinaten der rechten, oberen Ecke *)
      koordinatenpaare[3] := rechtsx;
      koordinatenpaare[4] := rechtsy;
      gdos (kontrollfeld     , ascii_werte                , ascii_ausgabeparameter,
            koordinatenpaare , koordinatenausgabeparameter );
     end;
    begin
      (* Initialisierungswerte einstellen *)
      kontrollfeld[1] :=  1 ; (* Funktionsnummer *)
      kontrollfeld[2] :=  0 ; (* Keine Koordinaten *)
      kontrollfeld[4] := 10 ; (* Zehn Funktionsparameter *)
      (* Funktionsparameter *)
      ascii_werte[1]  := monitor;
      ascii_werte[2]  := linienart;
      ascii_werte[3]  := linienfarbe;
      ascii_werte[4]  := grafikcursorform;
      ascii_werte[5]  := grafikcursorfarbe;
      ascii_werte[6]  := zeichensatz;
      ascii_werte[7]  := textfarbe;
      ascii_werte[8]  := fuellart;
      ascii_werte[9]  := fuellmuster;
      ascii_werte[10] := fuellfarbe;
      (* Initialisierung des Ausgabegeraetes *)
      gdos (kontrollfeld    , ascii_werte                , ascii_ausgabeparameter,
            koordinatenpaare, koordinatenausgabeparameter);
      clg;
      locate  (29,1);
      print   ('Kleine GSX-Demonstration');
      locate  (29,2);
      print   ('========================');
      kasten  (10000,10000,25000,20000);
      repeat
      until keypressed;
    end.
```

Listing. Verwendung von GSX-Grafikbefehlen (Schluß)

Basic-Logeleien

Der Basic-Befehlssatz des CPC bietet die vier Befehle AND, OR, NOT und XOR für logische Verknüpfungen und unterstützt logische Vergleiche in Berechnungen und Zuweisungen. Lesen Sie in diesem Beitrag, wie Sie diese Elemente optimal einsetzen, um zukünftig schnellere und kürzere Basic-Programme zu schreiben.

Daß Computer eigentlich dumm sind, ist unter Programmierern kein Geheimnis. So besteht auch der CPC wie jeder andere Computer im Grunde genommen nur aus einer Ansammlung von vielen Schaltern. Zugegebenermaßen sind diese Schalter recht sinnvoll angeordnet, doch letztendlich sind sie nur zum Ein- und Ausschalten gemacht.

Aus diesem Grund arbeitet der Computer nur mit einer Ja-Nein- beziehungsweise Wahr-Unwahr-Logik, während die menschliche Logik wesentlich differenzierter entscheidet (wahrscheinlich, vielleicht, eventuell, kaum, etc.).

Doch die absoluten Entscheidungen von Maschinen reizten bereits im letzten Jahrhundert den englischen Mathematiker George Boole, sich über diese Art der Wahrheitsfindung Gedanken zu machen. Er tat es ausgiebig und schuf so das erste System einer Algebra unter Verwendung des Binär-Alphabets, bestehend aus den Zeichen »1« und »0« für »wahr« und »falsch«.

Daraus entwickelte er wiederum einen eigenen Formalismus zur Behandlung von Aussagenverknüpfungen. Mit dem Fortschreiten der Technik nutzten andere Mathematiker die von ihm geschaffenen Grundlagen und schufen eine mathematische Disziplin, die nach ihrem Erfinder benannte Boolesche Algebra.

Erst dadurch wurden Entwicklung und Konstruktion von elektronischen Rechenwerken, wie wir sie heute kennen, möglich, weshalb Boole auch als einer der Väter des Computers gilt.

Heute arbeiten alle Computer nach den Gesetzen der Booleschen Algebra. Diese Gesetze zu verstehen, ist für den Programmierer sehr wichtig, denn so wie die Grammatik zum Verständnis einer Sprache notwendig ist, ist auch das Verständnis der Denkweise von Computern für den Umgang mit diesen flinken Gesellen unerläßlich.

Ziel dieses Beitrags ist es, die Arbeitsweise Ihres Computers zu durchleuchten, und Ihnen so die Basis zu geben, Ihre Programme schneller, kürzer und effektiver zu schreiben.

Dabei spielt es im Prinzip keine Rolle, welcher Programmiersprache Sie sich bedienen. Alle nachfolgenden Beispiele sind jedoch in Basic geschrieben, damit Sie sie direkt nachvollziehen können.

Der zweite Teil dieses Beitrags beschreibt ausführlich den Aufbau eines interessanten Spiels, das durch die Anwendung logischer Gesetze und Vergleiche trotz Basic-Code sehr schnell und zugleich für seine Funktion sehr kurz ist.

Logik spart Zeit und Speicherplatz

Wie unterschiedlich die Geschwindigkeiten von Basic-Programmen bei gleicher Funktion sein können, demonstriert eindrucksvoll das folgende Beispiel.

Aufgabe ist es, der Variablen A 20000mal abwechselnd den Wert -1 und 0 zuzuweisen. Zur Zeitkontrolle dient die TIME-Funktion des CPC.

Bei konventioneller Programmierung entsteht folgendes Programm:

```
10 t=TIME
20 a=0
30 FOR b=1 TO 20000
40 a=a-1: IF a=-2 THEN a=0
50 NEXT b
60 t=TIME-t:PRINT t;"Einheiten
   benoetigt."
```

Dieses Programm benötigt für einen Durchlauf etwa 30000 Einheiten, das sind rund 100 Sekunden.

Wenn Sie nun Zeile 30 durch

```
30 a=-ABS(a+1)
```

ersetzen, sind es nur noch 20000 Zeiteinheiten, das heißt unter 70 Sekunden.

Versuchen Sie es noch mit

```
30 a=-a-1
```

und

```
30 a=NOT a
```

und Sie erhalten Werte um 18500 und 16000. Die Rechenzeit hat sich demnach fast halbiert, indem lediglich eine Zeile durch einen leistungsfähigeren Befehl ersetzt wurde.

Selbstverständlich richtet sich die Anwendung dieser Befehle auch noch nach anderen Kriterien, doch intensives Nachdenken lohnt sich fast immer.

Ist Ihnen das computergerechte logische Denken erst einmal in Fleisch und Blut übergegangen, dann gehen Sie damit um wie der Jongleur mit seinen Bällen.

Eine weitere Methode, Programme zu beschleunigen, ist inzwischen hinreichend bekannt: die Definition von Variablen, die nur ganzzahlige Werte annehmen, als Integer-Variablen (zum Beispiel »DEFINT a,b«).

In diesem Fall muß der Computer nur ein Doppelregister mit insgesamt 16 Bit berücksichtigen, und nicht mit mehreren Speicherstellen, Kommata und Exponenten rechnen.

In den folgenden Beispielen werden wir auch nur Variablen mit ganzzahligen Werten benutzen.

Am Anfang unserer Betrachtungen haben wir die Zahlen »1« und »0« als Symbole für »wahr« und »unwahr« genannt.

Doch warum erscheint dann beim Befehl »PRINT NOT 0« (der NOT-Befehl bildet das logische Gegenteil einer Zahl) der Wert -1, und nicht 1? Und warum sind als Integer-Werte lediglich die Zahlen von –32768 bis 32767 zugelassen?

Die Antwort liegt in der Computerdarstellung von Zahlen begründet. Mit 16 Bit lassen sich insgesamt nur $2^{16}=65536$ verschiedene Werte darstellen. Um jedoch auch die negativen Zahlen zu berücksichtigen, wird ein kleiner Trick benutzt und die 0 in die Mitte des Zahlenbereichs gesetzt.

Warum –1 statt 1?

Dazu wird das ganz links stehende, oberste Bit als Vorzeichen der Zahl betrachtet. Eine »0« signalisiert ein positives und eine »1« ein negatives Vorzeichen.

Auf diese Art und Weise kommt beim Invertieren die -1 zustande (den Vorgang, einen Wert in sein logisches Gegenteil zu verkehren, nennt man Invertieren). Der Computer dreht die 16 Bit einer Zahl einfach um, er invertiert sie. Bei 16 Nullen ergeben sich demnach 16 Einsen, hexadezimal ausgedrückt FFFF.

Wenn Sie »PRINT & FFFF« eingeben, sehen Sie selbst, wie der Computer den Wert in eine negative Zahl umformt. Diese Umformung entspricht dem Bilden des Einer-Komplements einer Zahl.

Jeder Vergleich von zwei ganzzahligen Werten, wie zum Beispiel in IF-THEN-Abfragen, ergibt beim CPC entweder 16 Nullen oder 16 Einsen.

Überlegen Sie einmal zur Probe, welchen Wert Ihr Computer beim Befehl »PRINT NOT 500« ausdruckt. Wenn Sie in der Dezimal-Binär- und Binär-Dezimal-Umrechnung schon bewandert sind, fällt Ihnen die Lösung nicht schwer. Ansonsten probieren Sie es einfach aus.

Um genauer zu erkennen, wie das Invertieren funktioniert, lassen Sie das folgende Programm zur Umrechnung von dezimalen in 16stellige Binärzahlen laufen.

```
10 DEFINT a-z
20 b=500
30 PRINT b;"= binaer &X";
   BIN$(b,16)
40 b=NOT b
50 PRINT b;"= binaer &X";
   BIN$(b,16)
```

Der Vergleich zeigt deutlich, wie der Computerbeim NOT-Befehl die einzelnen Bits invertiert.

Allerdings gibt es, wie im folgenden Beispiel, auch Überraschungen.

```
10 b=-501
20 IF b THEN PRINT b;"ist
   richtig."
30 IF NOT b THEN PRINT b;"
   ist falsch."
50 PRINT
60 PRINT"Was stimmt denn nun?"
```

Setzen Sie eine beliebige andere Zahl außer 0 und 1 ein, und Sie erhalten das gleiche Ergebnis, weil der Computer feststellt, daß nicht alle Bits entweder auf 0 oder 1 gesetzt sind. So registriert er, daß einzelne Bits übereinstimmen, andere wiederum nicht.

Wenn Sie die Variable b durch 0 oder -1 ersetzen, funktionieren die Vergleiche korrekt.

Um eine eindeutige Aussage zu erhalten, darf man nur ganzzahlige Werte miteinander vergleichen. Der Vergleich »a=b<0« weist der Variablen a beispielsweise den Wert -1 zu, wenn b gleich -500 ist, und den Wert 0, wenn b gleich 500 ist.

Was sind Aussagen?

Man unterscheidet in der Booleschen Algebra generell zwischen komplexen Aussagen und bitweisen Vergleichen.

Zu diesem Thema existiert viel weiterführende Literatur, und jeder Informatikstudent muß sich zwangsläufig mit diesem trockenen Thema auseinandersetzen. Umfangreiche Wahrheitstabellen sind neben dem Verknüpfen von Aussagen sein tägliches Brot.

Da Sie jedoch vermutlich kein Informatikstudent sind, wollen wir uns auf die wichtigsten Grundlagen beschränken. Sie sollten sich als Definition merken, daß Aussagen entweder wahre oder unwahre logische Sätze sind. Falls Sie jedoch mit Ihrem Wissen Freunde und Kollegen beeindrucken möchten, bemerken Sie leichthin, daß man unter einer Aussage die gedankliche Widerspiegelung eines Sachverhaltes der objektiven Realität versteht, und man wird vor Ihnen den Hut ziehen!

Gesprochene und geschriebene Sätze sowie mathematische und technische Formeln sind Formen von Aussagen. »X+3=7« ist beispielsweise solch eine Form.

Verknüpfungen von Aussagen treten in drei Grundformen auf, aus denen sich alle weiteren Verknüpfungen ableiten. Die erste Grundform ist die Konjunktion oder UND-Verknüpfung, die nur dann wahr ist, wenn alle Teilaussagen wahr sind und bei wenigstens einer unwahren Teilaussage ebenfalls unwahr ist.

Die ODER-Verknüpfung, auch Disjunktion genannt, ist wahr bei wenigstens einer wahren Teilaussage und nur unwahr, wenn alle Teilaussagen unwahr sind.

Die letzte Grundform ist die Verneinung oder Negation einer Aussage, die unwahr ist, wenn die Aussage wahr ist, und wahr, wenn die Aussage unwahr ist.

Wie in der »normalen« Arithmetik, existiert auch in der Aussage-Logik eine festgelegte Rangfolge der durchzuführenden Verknüpfungen. Schon in der Grundschule wird gelehrt, daß Punkt- vor Strichrechnung geht. Ähnliches gilt auch bei der Booleschen Algebra.

In komplexen Aussagen löst man zuerst die Klammern von innen nach außen auf und anschließend vorhandene Negationen. Dabei werden verneinte Aussagen negiert, um den ursprünglichen Zustand wiederherzustellen. Anschließend steht in der Reihenfolge die Berechnung der UND-Verknüpfungen und am Schluß die der ODER-Verknüpfung an.

Vielen Programmen sieht man die Unsicherheit des Autors an, wenn zum Beispiel Ausdrücke wie »X=(Y*2)+A-(X*A)« enthalten sind. In diesem Beispiel sind nämlich beide Klammern überflüssig. Grundsätzlich geht der Computer bei der Berechnung immer von links nach rechts vor, Rechnungen in Klammern werden zuerst durchgeführt, und die Punktrechnung hat Priorität vor der Strichrechnung.

Versuchen wir einmal als konkretes Beispiel die folgende Aussage computergerecht aufzubereiten »Ich gehe ins Kino, wenn ein interessanter Film läuft und noch Karten zu bekommen sind. Andernfalls lege ich mich ins Bett.«

Setzt man A für »ins Kino« und B für »ins Bett«, X für »Karten vorhanden« und Y für »Film interessant«, so ergibt sich die Aussage

```
A= X=-1 AND Y=-1: B = NOT A
```

oder noch kürzer

```
A= X AND Y:B= NOT A
```

Gleichgültig, welche Werte X und Y annehmen, A und B werden immer ein entgegengesetztes Ergebnis bringen.

Als Basic-Programm sieht der ganze Ausdruck folgendermaßen aus:

```
10 CLS
20 INPUT "Laeuft ein
   interessanter Film (J/N) ";Y$
30 INPUT "Sind noch Karten
   vorhanden (J/N) ";X$
40 Y$=UPPER$(LEFT$(Y$,1)):
   X$=UPPER$(LEFT$(X$,1))
50 PRINT
60 A=X$="J" AND Y$="J"
70 B=NOT A
80 IF A THEN PRINT "Ich gehe ins
   Kino"
90 IF B THEN PRINT "Ich lege mich
   ins Bett"
100 WHILE INKEY$="":WEND:RUN
```

Setzen und Löschen Bit für Bit

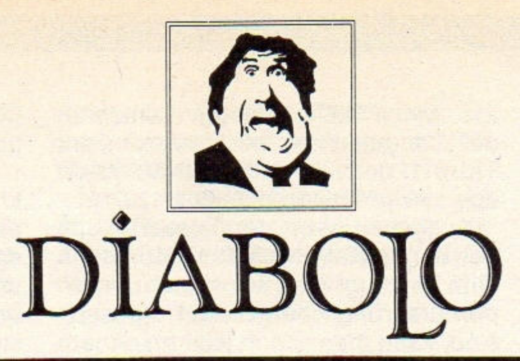
Eine computergerechte Aussage besteht letztendlich aus einzelnen Wörtern (zu je 16 Bit), diese aus zwei Byte und jedes Byte wiederum aus 8 Bit, von denen jedes den Zustand 0 und 1 annehmen kann. Zusätzlich läßt sich jedes Byte in zwei Teile zu je 4 Bit aufteilen, die als Nibble bezeichnet werden.

Es spielt eine große Rolle, welche Wertigkeit ein Bit innerhalb des Bytes einnimmt. Steht es ganz rechts, so hat es den Wert $2^0=1$ (sofern es gesetzt ist), steht es dagegen ganz links, so beträgt der Wert immerhin $2^7=128$.

Die Wertigkeit eines Bits ist damit jeweils eine Potenz von 2, so wie in dem uns vertrauten Dezimalsystem die Wertigkeit einer Stelle innerhalb einer Zahl eine Potenz von 10 ist.

Rückt man im Dezimalsystem die 1 um eine Stelle nach links und füllt die leere rechte Stelle mit einer 0 auf, so hat man die 1 mit 10 multipliziert und das Ergebnis 10 erhalten.

Im Binärsystem ist es nicht anders. Rückt man eine rechts stehende 1 um eine Stelle nach links, so hat man die Zahl mit 2 multipliziert und produziert damit eine 2, im Binärsystem dargestellt als 10. Eine weitere Verschiebung um eine Stelle bedeutet eine



★ Der Versand mit den teuflischen Preisen! ★

	Cass.	Disk.
Academy (Tau Ceti II)	DM 25.90	37.90
ACE	DM 25.90	37.90
ACE of ACEs	DM 25.90	37.90
Aliens	DM 25.90	37.90
Arkanoid	DM 25.90	37.90
Auf Wiedersehen Monty	DM 25.90	37.90
Avenger	DM 25.90	37.90
Ballblazer	DM 25.90	37.90
Big Trouble in little China	DM 25.90	37.90
BMX Simulator	DM 9.90	—.—
Bombjack II	DM 24.90	35.90
Break Thru	DM 25.90	37.90
Christal Castle	DM 25.90	37.90
Copout	DM 25.90	—.—
Dragons Lair	DM 19.90	33.90
Dragons Lair II	DM 25.90	37.90
Druid	DM 19.90	37.90
Enduro Racer	DM 25.90	37.90
Explorer	DM 25.90	37.90
Gauntlet	DM 24.90	33.90
Ghosts'n Goblins	DM 25.90	37.90
Grand Prix	DM 9.90	—.—
Hacker II	DM 25.90	37.90
Head over Heels	DM 25.90	37.90
Hit-Pack	DM 25.90	37.90
Howard the Duck	DM 25.90	37.90
Ikari Warrior	DM 25.90	—.—
Indoor Sports	DM 25.90	37.90
Infiltrator	DM 24.90	33.90

	Cass.	Disk.
Jail Break	DM 19.90	29.90
Konamis Coin up Hits	DM 25.90	37.90
Leaderboard	DM 25.90	37.90
Legend of Kage	DM 19.90	29.90
Lightforce	DM 25.90	37.90
Mercenary	DM 25.90	—.—
Marble Madness	DM 24.90	37.90
Masterchess	DM 9.90	—.—
Muncher (PacMan)	DM —.—	25.90
Puzzle (R+E Software)	DM —.—	29.00
Rescue on Fractalus	DM —.—	27.90
Sailing	DM 25.90	37.90
Scooby Doo	DM —.—	29.90
Sentinel	DM 25.90	37.90
Shaolin's Road	DM 25.90	37.90
Shockway Rider	DM 19.90	29.90
Space Harrier	DM 29.90	39.90
Spy vs Spy II	DM 25.90	37.90
Starglider	DM 33.90	44.90
Starrider II	DM 25.90	37.90
Strike Force Cobra	DM 25.90	37.90
Supercycle	DM 25.90	37.90
Superstory (nur 464)	DM 19.90	—.—
Tempest	DM 25.90	37.90
Top Gun	DM 24.90	—.—
Trailblazer	DM 25.90	37.90
Xevius	DM 25.90	37.90
Yie ar Kung Fu II	DM 25.90	37.90

NEU NEU NEU NEU

	Cass.	Disk.
Barbarian	DM 25.90	37.90
Bubbler	DM 25.90	37.90
Dogfight 2187	DM 25.90	37.90
Hydrofool	DM 25.90	37.90
Leviathan	DM 25.90	37.90
Livingstone	DM 25.90	37.90
Mag Max	DM 25.90	37.90
Mario Brothers	DM 25.90	37.90
Metrocross	DM 25.90	37.90
Nemesis	DM 25.90	37.90
Palitron	DM 25.90	37.90
Pulsator	DM 25.90	37.90
Thing bounces back	DM 25.90	37.90
Worldgames	DM 25.90	37.90

S★A★M★P★L★E★R★S

Elite

Hit Pack
Airwolf, Bombjack, C., Frank Bruno's Boxing
C 25.90 D 37.90

Imagine

Konami's Coin-Up Hits
Hypersports, G.B., Ping Pong, Mickie, Yie ar Kung Fu
C 25.90 D 37.90

Six Pack

7 auf einen Streich
Antiriad, Jet Set Willy II, Scooby Doo, Split Personalities, Fighting Warrior, Bomb Jack, Duet
C 29.90 D 39.90

Mikro Gen

Classic Collection No. 1
Stainless Steel, Frost Byte, Pyjamarama, Battle of the Planets
C 25.90 D 37.90

R+E Software

The Player's Dream I
Darts, Senso, Showdown, Jump Over, Pingo, Zentus, Steinschlag, Centibug, Jolly Jumper, Pyramide
C 19.90 D 24.90

R+E Software

The Player's Dream II
Sepp im Hochhaus, Minigolf, Tennis, Astronaut, Suicide Squad, Royal Flush, Flowers, Roulette, Buggy Blaster
C 19.90 D 24.90

Diabolo-Anwenderprogramme ● 3 for 1

Data Base
(Datenverwaltungsprogramm)

ZEN
(Z 80-Assembler)

Logo
(Turtle-Graphic-Interpreter)

DM 25.-

Diese 3 Programme gibt es nur für den CPC 464 und nur auf Cassette!

Solange Vorrat reicht!

Software-Bestellschein

Ich bestelle aus dem Diabolo-Versand folgende Software:

hc S

Anzahl	Titel	Gesamt-preis

Ich wünsche folgende Bezahlung:
☐ Nachnahme (zuzüglich 5.70 DM Versandkosten)
☐ Vorauskasse (zuzüglich 3 DM Versandkosten, ab 100 DM Bestellwert versandkostenfrei)
Bei Vorauskasse bitte Scheck beilegen

Name des Bestellers

Anschrift

PLZ/Ort

Datum/Unterschrift

Coupon ausschneiden, auf Postkarte kleben und einsenden an:
Diabolo-Versand, Postfach 16 40, 7518 Bretten.
Eine Abteilung des Verlags Rätz-Eberle GdbR.

abermalige Multiplikation und als Ergebnis den Wert 4, dargestellt als 100.

Doch was fängt man nun mit diesen Erkenntnissen an? In Basic gibt es hier nützliche Anwendungsgebiete, in denen sich mit Hilfe von Bitmanipulationen optimale Ergebnisse erzielen lassen. Das bekannteste Beispiel ist die Joystickabfrage mit dem JOY-Befehl.

Logik im Labyrinth

Doch es gibt weitere Einsatzbeispiele, von denen wir im folgenden eines exemplarisch behandeln wollen. Betrachten wir die Konstruktion eines Labyrinths. Dieses Gebilde besteht ursprünglich aus vielen quadratischen, geschlossenen Zellen. In Adventure-Programmen symbolisieren meistens die vier Himmelsrichtungen die Wände, doch das Prinzip bleibt gleich.

Es bietet sich nun an, ein Nibble für einen Raum zu nehmen, so daß sogar noch vier Bit eines jeden Bytes für weitere Funktionen verbleiben.

Doch bleiben wir beim Nibble. Jedes Bit stellt eine Wand oder Tür dar, die entweder geschlossen oder geöffnet ist. Das bedeutet nichts anderes als ein gesetztes oder gelöschtes Bit. Die Wertigkeit stellt dabei die Himmelsrichtung dar. In unserem Beispiel soll 1=Norden (Bit 0), 2=Osten (Bit 1), 4=Süden (Bit 2) und 8=Westen (Bit 3) sein. Sind an allen vier Seiten Wände und ist damit kein Ausgang vorhanden, so ergibt sich der Wert 1+2+4+8=16.

Stellen Sie sich vor, Sie finden nach Ausschalten des giftigen Zwergs den dringend gesuchten gläsernen Schlüssel und können damit die südliche Tür öffnen. Es bleibt nur noch das Problem, das entsprechende Bit (Bit 2) zu löschen, um einen freien Ausgang zu erhalten. Dies läßt sich zwar erreichen, indem man die Wertigkeit von Bit 2 von 16 subtrahiert (Ergebnis: 12), doch weil wir bei einem Programmlauf nicht immer ohne weiteres beurteilen können, ob die Tür nicht bereits offen ist, kann die simple Subtraktion unter Umständen zu Komplikationen führen.

Es empfiehlt sich der logische Vergleich, um das Bit (falls gesetzt) zu löschen. Befindet sich der Bytewert des Nibble in der Variablen a, dann erzeugt

```
a= a AND (255-4)
```

das gewünschte Ergebnis.

Dies läßt sich an einem praktischen Beispiel leicht überprüfen.

Nehmen wir an, a erhält den Wert 215. Mit »PRINT BIN$(a)« berechnet der Computer das binäre Äquivalent 11010111 und mit »PRINTBIN$(255-4)« den zweiten binären Wert 11111011.

Ausgehend von der Tatsache, daß bei der UND-Verknüpfung nur dann eine 1 erzeugt wird, wenn auch die beiden ursprünglichen Bit auf 1 gesetzt sind, kann man schon jetzt erkennen, daß sich beim Vergleichen die dritte 1 von rechts in eine Null verwandeln wird, alle anderen Bit im unteren Nibble dagegen ihren Wert beibehalten. Die Bestätigung dieser Überlegung erhalten Sie mit »PRINT BIN$(a AND (255-4))«. Das Resultat ist - wie erwartet - 11010011 und das gewünschte Bit 2 ist gelöscht.

Wenn Sie jetzt die Tür nach Süden wieder schließen möchten, muß auch das Bit 2 wieder gesetzt werden. Dies geschieht ebenso einfach mit der ODER-Verknüpfung. Der Befehl »PRINT BIN$(a OR &X100)« setzt Bit 2 und ergibt wieder 11010111 beziehungsweise 215.

Wäre im letzteren Fall Bit 2 bereits gesetzt gewesen, so hätte sich nichts geändert, während die simple Addition von 4 zu dem unteren Nibble dagegen ein falsches Ergebnis erzeugt hätte.

15 Monster in 4 Bit

Bleiben wir bei dem Adventure-Beispiel und erinnern wir uns, daß Integerzahlen insgesamt 16 Bit umfassen. Uns bleiben demnach noch 12 auf der linken Seite, mit denen wir, bezogen auf einen Raum, etwas anfangen können. Zweigen wir 7 Bit ab, so lassen sich damit immerhin $2^7=128$ Räume numerieren, und 4 weitere Bit ergeben 15 verschiedene Monster und einmal kein Monster, oder 15 verschiedene Gegenstände und einmal kein Gegenstand, oder Monster und Gegenstände gemischt.

Das letzte, ganz links stehende Bit kann man dann noch als Lichtschalter (Raum bekannt/unbekannt) oder ähnliches benutzen. Zum Überprüfen dieses Bits genügt eine einfache Abfrage, ob der Integerwert positiv oder negativ, also kleiner als 0 oder größer gleich 0 ist.

Je 7 Bit aus den 16 Bit einer Integerzahl reichen also, um 128 Räume vollständig zu beschreiben! Da Adventure-Programme bekanntlich äußerst speicherplatzfressend sind, ist das vorgestellte Verfahren eine exzellente Methode, um Informationen über das Gebiet, in dem ein Adventure spielt, speicherplatzsparend abzulegen.

Der Vollständigkeit halber sei auch noch die logische XOR-Verknüpfung (XOR=Exklusiv-Oder) erwähnt, die ganze Bitfolgen umkehren kann.

Das Ergebnis einer XOR-Verknüpfung ist wahr, wenn die eine Teilaussage wahr und die andere Teilaussage unwahr ist. Das Ergebnis ist unwahr, wenn beide Teilaussagen unwahr oder beide Teilaussagen wahr sind.

In unserem Abenteuerbeispiel würde XOR wie ein Wechselschalter wirken. Wenn Sie den Befehl

```
PRINT BIN$(A XOR 4)
```

mehrmals ausprobieren, erkennen Sie am besten die Wirkung der XOR-Verknüpfung.

Daß der CPC ebenfalls von der Bitumschaltung Gebrauch macht, ist leicht an der Groß- und Kleinschreibung zu erkennen. Der ASCII-Code ist so aufgebaut, daß der Unterschied zwischen großen und kleinen Buchstaben genau 32 beträgt, was wiederum dem Bit 5 mit der Wertigkeit von $2^5=32$ entspricht. Der Befehl

```
PRINT CHR$(ASC("A") XOR 32)
```

macht dies deutlich. Umgekehrt funktioniert es natürlich ebenfalls.

Doch der ASCII-Code läßt sich auch für andere Zwecke einsetzen. Müssen in einem Programm beispielsweise viele Zahlen gespeichert werden, ist es angebracht, diese zu komprimieren. Neben der Platzeinsparung werden sie gleichzeitig codiert und sind dadurch nicht mehr für jedermann lesbar.

Betrachtet man den ASCII-Zeichensatz, erkennt man, daß die Zeichen von 35 bis 255 benutzt werden dürfen, ohne als Steuerzeichen Konflikte zu verursachen. Beschränken wir uns auf den Bereich von 128 bis 255, so ist es noch einfacher. Die Zahl 3985245 soll beispielsweise verschlüsselt werden. Ein kleines Programm nimmt uns diese Arbeit ab.

```
100 a=3985245
110 a$=STR$(a)
120 a$=MID$(a$,1+LEN(a$)MOD 2)
130 c$="":FOR b=1 TO LEN(a$)
    STEP 2
140 c$=c$+CHR$(VAL(MID$(a$,b,2))
    XOR 128)
150 NEXT b
160 PRINT c$
```

Das Ergebnis unterscheidet sich vollständig vom Ausgangswert und läßt keinerlei Rückschlüsse darauf zu. Natürlich ist statt 128 auch jeder andere Wert bis 255 erlaubt, denn der zu codierende Wert kann nie kleiner als 0 und größer als 99 sein. Genauso simpel läßt sich die erzeugte Zeichenfolge wieder in die ursprüngliche Zahl zurückverwandeln.

```
200 a$=""
210 FOR b=1 TO LEN(c$)
220 a=ASC(MID$(c$,b,1)) XOR 128
```

```
230 a$=a$+MID$(STR$(a),2)
240 NEXT b
250 a=VAL(a$)
260 PRINT a
```

Das Basic des CPC macht die Umrechnung von dezimal in binär und von binär in dezimal leicht, weil es dazu die Befehle BIN$ und PRINT &X zur Verfügung stellt. Doch wenn man sich beispielsweise gerade in einem Programm befindet und etwas austesten möchte, muß man dieses Programm erst verlassen, um einen Wert umrechnen zu können.

Deshalb im folgenden das Verfahren, wie man eine Dezimalzahl in einen binären Wert und eine Binärzahl in den dezimalen Wert umrechnet.

In unserem Beispiel soll die Zahl 179 umgewandelt werden. Sie schreiben:

```
179/2 =  89, Rest 1
 89/2 =  44, Rest 1
 44/2 =  22, Rest 0
 22/2 =  11, Rest 0
 11/2 =   5, Rest 1
  5/2 =   2, Rest 1
  2/2 =   1, Rest 0
  1/2 =   0, Rest 1
```

Die Werte, die den Rest der Division angeben, schreiben Sie nun von unten nach oben hintereinander – und fertig ist der binäre Wert von 179: 10110011.

Noch einfacher funktioniert die Umrechnung von binär in dezimal. Gehen Sie dazu von rechts nach links vor.

$1 \times 2^0 = 1$
$1 \times 2^1 = 2$
$0 \times 2^2 = 0$
$0 \times 2^3 = 0$
$1 \times 2^4 = 16$
$1 \times 2^5 = 32$
$0 \times 2^6 = 0$
$1 \times 2^7 = 128$

Summe: 179

Haben Sie erst einmal das grundsätzliche Prinzip verstanden, dann können Sie in jedes andere Zahlensystem umrechnen.

Im jedem Maschinensprache-Programmierer geläufigen Hexadezimalsystem sieht die Umrechnung folgendermaßen aus:

```
179/16 = 11 Rest  3 = 3
 11/16 =  0 Rest 11 = B
```

Der hexadezimale Wert von 179 beträgt demnach B3.

Die Rückrechnung in das Dezimalsystem:

$3 \times 16^0 = 3$
$11 \times 16^1 = 176$

Summe: 179

Wenn Sie mit Nibbles arbeiten, können Sie einen hexadezimalen Wert direkt in eine Binärzahl umsetzen. Die Ziffer einer hexadezimalen Zahl besteht aus 4 Bit, da sie Werte von 0 bis F, das heißt $16=2^4$ verschiedene Zustände annehmen kann. Es besteht folgende Beziehung zwischen den hexadezimalen Ziffern und den Binärwerten:

Hex	Bin	Hex	Bin
0	0000	8	1000
1	0001	9	1001
2	0010	A	1010
3	0011	B	1011
4	0100	C	1100
5	0101	D	1101
6	0110	E	1110
7	0111	F	1111

Der binäre Wert von B beträgt demnach 1011, und der binäre Wert von 3 ist 0011. Deshalb entspricht die Hexadezimalzahl B3 dem binären Wert 10110011. Die Umrechnung von binär in hexadezimal ist genauso einfach.

(Dietmar Schulze/ma)

Boolesche Algebra im Weltraum

Nachdem wir im vorangegangenen Beitrag die Grundlagen zur effektiven Programmierung mit logischen Vergleichen und Verknüpfungen behandelten, folgt nun die Entwicklung eines pfiffigen Action-Spiels, in dem die besprochenen Verfahren praktische Anwendung finden.

Im ersten Teil dieses Beitrags haben wir Ihnen ziemlich viel Theorie zugemutet. Als Belohnung für Ihre Aufmerksamkeit wollen wir uns nun ganz auf die Praxis konzentrieren und anhand eines rasanten Action-Spiels zeigen, wie man mit logischen Vergleichen und Verknüpfungen effektiv programmiert, so daß sogar in Basic hohe Verarbeitungsgeschwindigkeiten zustande kommen.

Bei dem Spiel »Spacetrap« gilt es, mit einem defekten Raumschiff aus einem durch Energiefelder und Robotschiffe gesicherten Raumsektor zu entkommen. Geschossen wird nicht, vielmehr müssen Sie die feindlichen Schiffe in ihre eigenen Fallen locken und dadurch vernichten. Im Laufe des Spiels schickt der Feind immer größere Geschwader auf das Schlachtfeld.

Das Programm bietet eine ausführliche Beschreibung. Es ist so gestaltet, daß die vielfältigen Fähigkeiten des Locomotive-Basic optimal genutzt werden. Manche der verwendeten Techniken dienen jedoch nur zur Demonstration und lassen sich auch anders formulieren.

Speichern Sie bitte die Listingteile, die im folgenden vorgestellt werden, entweder einzeln und verbinden Sie sie mit dem MERGE-Befehl, oder tippen Sie das Listing in einem Stück ein.

Die ersten Zeilen des Programms legen einige Anfangswerte fest und die letzten Zeilen definieren die Zeichen, die im Spiel verwendet werden (Listing 1).

Dann wird der Bildschirm mit Überschrift aufgebaut. Für uns ist die Behandlung der Farbenvariable <col> interessant, die innerhalb der Schleife je nach Wert der Zählervariablen <i> auf 0 oder 1 gesetzt wird und dadurch das Streifenmuster des Schrifthintergrundes erzeugt. Der Wert von <col> ergibt sich aus dem Aussagenvergleich von gerader und ungerader Zahl, der direkt zum Berechnen von 0 oder 1 benutzt wird. Der gleiche Effekt läßt sich auch mit »col=i MOD 2« erreichen.

Der Schriftzug selbst wird zum Erzielen einer Schattenwirkung zweimal im Transparentmodus gedruckt und löscht dadurch auch nicht den Hintergrund. Der übrige Bildaufbau geschieht mit Strings und durch Benutzung der Schleifenvariablen als Positionszeiger für den LOCATE-Befehl (Listing 2).

Der folgende Programmteil legt die Anfangswerte für den Programmstart fest und baut das interne Spielfeld im

eigens dafür reservierten Speicherbereich auf. Das Speichern in Feldvariablen wäre natürlich auch möglich gewesen, aber die PEEK- und POKE-Befehle sind in der Regel schneller.

Sämtliche Felder werden zuerst mit dem ASCII-Wert 32 (für das Leerzeichen) belegt. Während dieses Vorganges wird zur Verkürzung der Wartezeit der Bereich des Spielfelds auf dem Bildschirm mit Sternchen gefüllt.

Die Variablen <ef$>, <fs$> und <es$> mit den Werten <ef>, <fs> und <es> sind die ASCII-Codes für Energiefeld, Robotschiffe und das eigene Raumschiff. Die Variable <er$> dient zum Löschen einer Zeile.

Ab und zu bemerken Sie bei Tastaturabfragen den Ausdruck »rd=RND«. Diese Zuweisung zählt lediglich die Zufallszahlen weiter, um die feste Pseudozufallsreihe des Computers zu unterbrechen und dadurch neue Anfangswerte festzulegen.

Wird die Ausgabe der Spielanleitung gewünscht, verzweigt das Programm zu einem entsprechenden Unterprogramm. Dabei rollt ein Vorhang aus farbigen Streifen herab, der wiederum durch die Schrift der Anleitung verdrängt wird. Farbwerte, Textzeilen, Abstände und Pausen sind in den DATA-Zeilen mit enthalten. Eine Schleife liest diese Werte ein und zentriert den Text auf dem Bildschirm.

Zum Abschluß erfolgt die Frage nach der Schwierigkeitsstufe, die die Anfangsgeschwindigkeit des Spiels bestimmt, und das eigene Schiff wird auf das Spielfeld gesetzt (Listing 3).

Jetzt müssen nur noch die feindlichen Energiefelder und Robotschiffe verteilt, sowie die Werte für Schwierigkeitsstufe, Spielpunkte etc. auf dem Bildschirm eingetragen werden (Listing 4).

Dann geht es endgültig los! Der Betriebssystemaufruf CALL &BB18 bringt die Angreifer in ihre Wartestellung. Sobald Sie nun eine Taste drücken, beginnt das Spiel und Sie sind auf der Flucht. Unerbittlich rücken die Roboterschiffe an. Deren Richtung wird dabei durch die Koordinaten des eigenen Raumschiffes bestimmt.

Die Zeilen 1000 bis 1220 bilden die Hauptschleife des Programms. Die Tastaturabfrage ist kurz, aber effektiv konstruiert. Es werden der Joystick, die Cursortasten sowie die Tasten <Q>,<A>, <O> und <P> abgefragt und die Werte für jede Bewegungsrichtung addiert.

Der INKEY-Befehl ist dabei sehr nützlich, weil er sich leicht auf »wahr«oder »falsch« abfragen läßt. Ist keine Taste gedrückt, dann ergibt sich bei der Addition die Summe -3. Gleichgültig, welche Taste nun mit oder ohne <SHIFT> betätigt wird, der Wert bewegt sich immer im Bereich größer als -3 und die Additionsvariablen x für die Horizontal- sowie y für die Vertikalposition ändern sich entsprechend. Ist keine Taste gedrückt, bleiben die Ergebnisse der logischen Aussagen 0, und eine Bewegung findet nicht statt.

Die Joystickabfrage nach dem Muster »IF JOY(0)=1 THEN« scheint nur auf den ersten Blick richtig, erweist sich aber schnell als trügerisch. Jeder Joystick läßt sich nämlich auch schräg betätigen, so daß zwei Kontakte gleichzeitig geschlossen sind und sich die Werte folglich addieren. In diesen Fällen gibt die JOY-Funktion die Werte 5 oder 9 aus, und die Abfrage ergibt »falsch«.

Im erwähnten Beispiel müssen die überflüssigen Bit ausgeblendet werden, damit die Abfrage funktioniert. »IF JOY(0) AND 1« scheint vernünftig, doch arbeitet diese Befehlsfolge auch wirklich korrekt?

Wieder nicht ganz, denn, steht vor dieser Abfrage beispielsweise »IF JOY(0) AND 4« und Sie haben zufällig nach vorwärts und links gedrückt, dann ist gleich die erste Bedingung erfüllt.

Als bester Weg erweist sich die Abspeicherung der Ergebnisse in Additionsvariablen.

```
X=((JOY(0) AND 4)=4) OR -((JOY(0) AND 8)=8):
Y=((JOY(0) AND 1)=1) OR -((JOY(0) AND 2)=2)
```

Durch diese Befehlsfolge erhalten beide Richtungs-Variablen die korrekten Werte 1, -1 oder 0, mit denen dann weitergerechnet werden kann.

Doch zurück zu unserem Programm. Die Variablen <eh> und <ev> enthalten die alten Positionen Ihres Raumschiffs. Zeile 1140 mit dem Befehl ON GOTO setzt das Programm bei wahrem Vergleich im Kollisions-Programmteil, ansonsten beim NEXT-Befehl in Zeile 1200 fort.

Wie Sie feststellen werden, erlaubt es diese Art der Programmierung, die Steuerschleife kurz zu halten und sie bewirkt eine präzise und schnelle Reaktion auf die Joystick- oder Tastatureingabe – eine Grundvoraussetzung bei eimem Reaktionsspiel (Listing 5).

Der Verzicht auf Unterbrechungen wirkt sich positiv aus. Nur bei einer Bewegung wird überprüft, ob Kollisionen stattgefunden haben und deshalb weitere Aktionen erforderlich sind. Dadurch wird immer nur das Nötigste ausgeführt. Erst wenn eine Kollision wirklich stattfindet, werden weitere Unterprogramme aufgerufen.

Die vielen GOSUB-Befehle sind notwendig, um aus vielen Routinen nur die gerade aktuellen auszuwählen und möglichst schnell zur Hauptroutine zurückkehren zu können. Außerdem wird dadurch das Programm flexibler. Der Preis, den man dafür zu zahlen hat, ist unter Umständen ein schwierigeres Verständnis der Programmfunktionen (Listing 6).

Zum Schluß die Bedeutung der wichtigsten Variablen im Programm:

- sp = senkrechte Position
- wp = waagrechte Position
- fp! = Feldadresse im Speicher
- h! = HIMEM
- q = Energiefelder, die beim nächsten Durchgang ersetzt werden müssen
- f = Anzahl der verbleibenden Robotschiffe
- c = Positionsnummer eines Robotschiffs

(Dietmar Schulze/ma)

◄ **So präsentiert sich »Spacetrap« auf dem Bildschirm**

Steckbrief	
Programm:	Spacetrap
Computer:	CPC 464/664/6128
Checksummer:	Explora
Datenträger:	Kassette, Diskette

```
100 '***********************         [44E8]
110 '*** Space Trap 1.0 ***          [03CE]
120 '***********************         [E6EC]
130 '                                [DFB6]
140 CLEAR:DEFINT a-z:MODE 1:IF HIMEM>425
    00 THEN SYMBOL AFTER 129:MEMORY HIME
    M-1000:GOSUB 10040'==> eigene Zeiche
    n definieren und Speicherbereich sic
    hern                             [C520]
10000 '==============================
                                     [512A]
10010 '*** Data fuer eigene Zeichen ****
                                     [FCBC]
10020 '==============================
                                     [012E]
10030 '                              [1176]
10040 RESTORE 10060                  [BDA0]
10050 READ z:IF z=256 THEN RETURN    [9EB0]
10060 FOR i=0 TO 7:READ z(i):NEXT    [DB62]
10070 SYMBOL z,z(0),z(1),z(2),z(3),z(4),
      z(5),z(6),z(7):GOTO 10050      [EDCA]
10080 DATA &C0,&3C,&66,&FF,&99,&3C,&66,&
      DB,&C3                         [6A74]
10090 DATA &C1,&18,&7E,&66,&DB,&DB,&66,&
      7E,&18                         [5858]
10100 DATA &C2,&6E,&A5,&FF,&BC,&3D,&FF,&
      A5,&76                         [C7BE]
10110 DATA &C3,&24,&5A,&DB,&FF,&FF,&DB,&
      5A,&24                         [6FAA]
10120 DATA &C4,&66,&C3,&A5,&18,&18,&A5,&
      C3,&66                         [E0FA]
10130 DATA &C5,&7C,&11,&99,&BF,&FD,&99,&
      88,&3E                         [066E]
10140 DATA 198,0,24,48,126,12,56,48,0  [72AE]
10150 DATA &F3,&6E,&A5,&FF,&BC,&3D,&FF,&
      A5,&76                         [C4D0]
10160 DATA &F4,&BD,&42,&99,&A5,&A5,&99,&
      42,&BD                         [BC54]
10170 DATA &F5,&42,&BD,&66,&5A,&5A,&66,&
      BD,&42                         [F040]
10180 DATA &F6,&0,&3C,&66,&5A,&5A,&66,&3
      C,&0                           [402C]
10190 DATA &F7,&0,&42,&18,&24,&24,&18,&4
      2,&0                           [F5A4]
10200 DATA &F8,&0,&0,&0,&18,&18,&0,&0,&0
                                     [5AE6]
10210 DATA &F9,0,0,0,0,0,0,0,0       [F0A6]
10220 DATA 256,0,0,0,0,0,0,0,0       [E398]
```

Listing 1. Der Definitionsteil von »Spacetrap«

```
150 ue$="******<2>S P A C E - T R A P<2>
    ******"                          [F53A]
160 SPEED INK 4,3:INK 0,0:BORDER 0:INK 1
    ,24:INK 2,20:INK 3,8             [9976]
170 ef$=CHR$(198):GOSUB 5040         [93F6]
180 '                                [06C0]
5000 '==========================     [4A70]
5010 '*  Aufbau der Ueberschrift *  [D0EC]
5020 '==========================     [F074]
5030 '                               [BC1E]
5040 MODE 1:WINDOW 1,40,5,24:WINDOW#1,1,
     40,1,3:WINDOW#2,1,40,4,4:WINDOW#3,1
     ,40,25,25                       [C5C6]
5050 PEN#1,0:PAPER#1,2               [10C2]
5060 FOR i=1 TO 20:col=-(INT(i/2)<>i/2)+
     1-(i=20):PLOT 0,399-i*2,col:DRAW 63
     9,399-i*2:NEXT                  [9C92]
5070 LOCATE#1,4,2:PRINT#1,CHR$(22)CHR$(1
     )ue$;CHR$(22)CHR$(0);:PRINT CHR$(23
     )CHR$(3);:x=32+12:y=399-14:TAG:PLOT
      x,y,3:PRINT ue$;:TAGOFF:PRINT CHR$
     (23)CHR$(0);                    [06C0]
5080 FOR i=0 TO 1:PLOT 4-i*2,393+i*2,i*3
     :DRAWR 633,0:DRAWR 0,-32:DRAWR -633
     ,0:DRAWR 0,32:NEXT              [BFEE]
5090 PEN 3:FOR i=0 TO 1:LOCATE 1,1+19*i:
     PRINT STRING$(40,ef$);:NEXT     [AAFA]
5100 FOR i=2 TO 20:FOR j=0 TO 1:LOCATE 1
     +j*39,i:PRINT ef$;:NEXT j,i:WINDOW
     2,39,6,23                       [47A2]
5110 PEN#2,1:LOCATE#2,1,1:PRINT#2," Bonu
     s:"SPC(9)"Level:"SPC(4)"Score:";:PE
     N#3,1:PRINT#3,"<2>HI-SCORE:"SPC(16)
     "SHIPS:";:RETURN                [4CF2]
5120 LOCATE#2,23,1:PRINT#2,USING"##";lev
     :LOCATE#2,34,1:PRINT#2,USING"####0"
     ;b1                             [3E20]
5130 LOCATE#2,9,1:PEN#2,2:PRINT#2,USING"
     ####0";b2\10:RETURN             [90F8]
5140 PEN#3,2:LOCATE#3,13,1:PRINT#3,USING
     "####0";MAX(b1,hi!);:LOCATE#3,35,1:
     PRINT#3,STRING$(MAX(0,r1),es)er$;  [E842]
5150 RETURN                          [8C96]
```

Listing 2. Das Begrüßungsbild von »Spacetrap«

```
500 '***********************         [D89C]
510 '*** Programmstart ***           [4818]
520 '***********************         [58A0]
530 '                                [07BE]
540 DEFINT a-z:h!=HIMEM+1:f1=3:r=4:r1=5:
    b1=0:q=30:e2=0                   [39AE]
550 GOSUB 5160'==> Anfangswerte      [DA14]
5160 ef$=CHR$(198):ef=198:er$=CHR$(18):e
     s$=CHR$(192):es=192:fs$=CHR$(196):f
     s=196                           [0C7E]
5170 fel=38*18:fp!=h!+fel:hi!=PEEK(fp!)*
     256+PEEK(fp!+1)                 [BA92]
5180 GOSUB 7300'==> Array loeschen   [C18E]
5190 LOCATE 2,6:PEN 2:PRINT"Wuenschen Si
     e eine Anleitung (J/N) ?";      [D540]
5200 i$="":WHILE i$<>"J"AND i$<>"N":i$=U
     PPER$(INKEY$):r=RND:WEND:PRINT i$  [0376]
5210 IF i$="J"THEN GOSUB 6040        [F350]
5220 LOCATE 2,9:PEN 2:PRINT"Schwierigkei
     tsgrad (0-9) ? ";               [A4EC]
5230 WHILE i$<"0"OR i$>"9":i$=INKEY$:rd=
     RND:WEND:p=VAL(i$)+1            [B61C]
5240 CLS:GOSUB 7230'==> Eigenes Schiff s
     etzen                           [1CBC]
5250 RETURN                          [C598]
5260 '                               [E728]
6000 '=====================          [571C]
6010 '* A n l e i t u n g *          [E992]
6020 '=====================          [DF20]
6030 '                               [C720]
6040 FOR i=0 TO 18*8-2:col=INT(RND*4):x=
     18:y=319-i*2:xe=INT(RND*(16*38)):PL
     OT x,y,col:DRAWR xe,0:xe=XPOS:PLOT
     639-x,y,3-col:DRAW xe,y:NEXT    [C4D8]
6050 RESTORE 6110                    [FEEE]
6060 READ b$,col:IF b$="xx"OR b$="yy" TH
     EN 6090                         [53E0]
6070 PEN col:LOCATE 1,18:PRINT CHR$(10)T
     AB(INT(39-LEN(b$))/2)b$;        [A4EA]
6080 GOTO 6060                       [A026]
6090 PEN 3:LOCATE 9,18:PRINT"Weiter mit
     LEERTASTE...";:PEN 1:WHILE INKEY$<>
     " ":WEND:IF b$="xx"THEN 6060    [623C]
6100 CLS:RETURN                      [5EC6]
6110 DATA "",1,"(c) 1986 by Dietmar Schu
     lze",2,,,"Katharinenhof 5",2,"5000<
     2>KOELN 1",2,,,"Tel:221/326121",2,,
     1,,1                            [7D00]
6120 DATA"=== GEFANGEN IM RAUM ===",3,,  [8DE4]
6130 DATA Umgeben von einem energiegelad
     enen,1,"Zaun sind Sie mit defektem
     HYPERSPACE-",1,"Triebwerk in einem
     unbekannten",1,"Raumsektor gelandet
     .",1,,1,,1,,1,xx,1              [EC16]
6140 DATA "",2,Zusaetzliche Energiefelde
     r sind eine,2,"weitere Gefahr, doch
      die groesste",2,Bedrohung kommt vo
     n den ploetzlich,2,auftauchenden Ro
     botschiffen.,2,,2               [918A]
6150 DATA Mit ihrem empfindlichen Ortung
     ssystem,2,registrieren sie jede Bew
     egung im,2,Sektor und greifen in di
     eser Richtung,2,sofort an.,2,,2  [D17C]
6160 DATA " Ihre einzige Rettung besteht
      darin,",2,die Angreifer in die Ene
     rgiefelder,2,zu locken. Dort werden
      sie mit diesen,2,zusammen vernicht
     et.,2,,2,Ein Aufprall auf ein eigen
     es Schiff,2,zerstoert sie ebenfalls
     .,2,,2,xx,2                     [9D18]
6170 DATA"",1,"=== S T E U E R U N G ===
     ",2,,1,"J o y s t i c k",1,,1,Curso
     rtasten,1,,1,oder,1,,1,"mit den Tas
     ten <Q>,<A>,<O>,<P>",1,,1       [1728]
6180 DATA"Sind alle Gegner erledigt, ers
     cheint",1,die naechste Flotte - zah
     lreicher und,1,schneller!,1,,1,Ihre
      Befreiung haengt nur von Ihrer,1,
     Geschicklichkeit ab... VIEL GLUECK
     !,1,,1,,1,yy,1                  [A7B2]
6190 '                               [A12E]
7000 '==================================
     =====================           [F3E4]
7010 '* locate wp,sp (waagerechte & senk
     rechte Position) *              [6C52]
7020 '==================================
```

Listing 3. In diesem Teil erfolgen die Voreinstellungen für den Spielbeginn

```
         =================                  [5DE8]
7030 '                                      [C622]
7040 wp=a MOD 38+1:sp=(a\38)+1:RETURN       [BA40]
7050 a=(wp-1)+(sp-1)*38:RETURN              [6C18]
7190 '================================      [B67C]
7200 '* Setzen des eigenes Schiffs *        [24BE]
7210 '================================      [966E]
7220 '                                      [9D24]
7230 PEN 2:r1=r1-1:IF r1<0 THEN 3040 ELS
     E GOSUB 5140                           [C57A]
7240 a=INT(RND*fel):IF PEEK(a+h!)<>32 TH
     EN 7240 ELSE POKE a+h!,es:GOSUB 704
     0:LOCATE wp,sp:PRINT es$;:eh=wp:ev=
     sp:GOSUB 5140:RETURN                   [68AA]
7250 '                                      [9C2A]
7260 '==================                    [BDC0]
7270 '* Array loeschen *                    [1036]
7280 '==================                    [2FC4]
7290 '                                      [2832]
7300 PEN 1:FOR a=0 TO fel-1:POKE h!+a,32
     :LOCATE INT(RND*38+1),INT(RND*18)+1
     :IF RND>0.85 THEN PRINT CHR$(144);E
     LSE PRINT" ";                          [3D7E]
7310 NEXT:RETURN                            [0C88]
7320 '                                      [BF26]
```

Listing 3. (Schluß)

```
560 GOSUB 7070:q=0:f=f1+2:f1=0'==> Energ
    iefelder setzen                         [2372]
570 b2=f*500:p=MAX(0,p-1):lev=10-p:GOSUB
     5120:GOSUB 5140                        [14F8]
580 IF f>24 THEN f=27:e2=e2+1:IF e2>3 TH
    EN 3040                                 [2A32]
590 GOSUB 7150'==> Robotschiffe setzen      [A282]
7060 '*** Setzen der Energiefelder ***      [BAEA]
7070 FOR b=1 TO q                           [5832]
7080 PEN 3:a=INT(RND*fel):IF PEEK(a+h!)<
     >32 THEN 7080 ELSE POKE a+h!,ef:GOS
     UB 7040:LOCATE wp,sp:PRINT ef$;        [8422]
7090 NEXT:PEN 1:RETURN                      [F96E]
7100 '                                      [C11E]
7110 '===========================           [62FE]
7120 '* Setzen der Robotschiffe *           [2EEE]
7130 '===========================           [2F02]
7140 '                                      [9D26]
7150 c=0:PEN 1:FOR b=1 TO f                 [150A]
7160 a=INT(RND*fel):IF PEEK(a+h!)<>32 TH
     EN 7160 ELSE POKE a+h!,fs:GOSUB 704
     0:LOCATE wp,sp:PRINT fs$;:POKE fp!+
     c,wp:POKE fp!+c+1,sp:c=c+2             [5BDE]
7170 NEXT:RETURN                            [1A90]
7180 '                                      [292E]
```

Listing 4. Hier werden die Robotschiffe verteilt

```
1000 CALL &BB18                             [BE60]
1010 '                                      [9212]
1020 '===================                   [2022]
1030 '* Tastaturabfrage *                   [AB7E]
1040 '===================                   [5426]
1050 '                                      [B61A]
1060 FOR pa=0 TO p:b2=MAX(b2-5,0)           [EE26]
1070 CALL &BB09                             [2B6E]
1080 x=eh:y=ev                              [D66C]
1090 '                                      [BA22]
1100 x=x+((INKEY(8)+INKEY(34)+INKEY(74))
     >-3)-((INKEY(1)+INKEY(27)+INKEY(75)
     )>-3)                                  [2312]
1110 '                                      [8E14]
1120 y=y+((INKEY(0)+INKEY(67)+INKEY(72))
     >-3)-((INKEY(2)+INKEY(69)+INKEY(73)
     )>-3)                                  [141C]
1130 '                                      [9018]
1140 ON 2+(x=eh AND y=ev)GOTO 1200,1150     [5B0C]
1150 IF((x<1)+(x>38)+(y<1)+(y>18))THEN 2
     050                                    [245C]
1160 wp=x:sp=y:GOSUB 7050                   [A6FA]
1170 IF PEEK(a+h!)<>32 THEN 2050            [1E70]
1180 PEN 2:LOCATE wp,sp:PRINT es$;:POKE
     a+h!,es                                [44CE]
1190 wp=eh:sp=ev:LOCATE wp,sp:PRINT" ";:
     GOSUB 7050:POKE a+h!,32:eh=x:ev=y      [83B6]
1200 NEXT pa                                [8526]
1210 GOSUB 2110:IF fg THEN fg=0:GOTO 100
     0                                      [AFF6]
1220 IF f>0 THEN 1060 ELSE CALL &BB09:b1
     =b1+b2/10:GOTO 560'==> keine Robots
     chiffe mehr                            [F48A]
```

Listing 5. Die Routine zur Tastaturabfrage ist der Hauptteil von »Spacetrap«

```
2000 '                                      [BA12]
2010 '===============                       [BC3A]
2020 '* Kollisionen *                       [E64C]
2030 '===============                       [D83E]
2040 '                                      [961A]
2050 pa=p+1:wp=eh:sp=ev:GOSUB 7050:GOSUB
      2220:GOTO 1000                        [4076]
2060 '                                      [981E]
2070 '==========================            [1D0A]
2080 '* Bewegung Robotschiffe *             [68AE]
2090 '==========================            [FD0E]
2100 '                                      [BC14]
2110 c=INT(RND*f)*2:wp=PEEK(fp!+c):sp=PE
     EK(fp!+c+1):GOSUB 7050:GOSUB 5130      [B5C6]
2120 x=wp:y=sp:wp=wp+(eh<x)-(eh>x):sp=sp
     +(ev<y)-(ev>y):GOSUB 7050              [9C2C]
2130 IF ev=sp AND eh=wp THEN GOSUB 2220:
     fg=1'==> Eigenes Schiff getroffen      [4DE0]
2140 pk=PEEK(h!+a):IF pk=ef THEN 2280 EL
     SE IF pk=fs THEN 2370                  [D1AA]
2150 POKE h!+a,fs:PEN 1:LOCATE wp,sp:PRI
     NT fs$;:POKE fp!+c,wp:POKE fp!+c+1,
     sp                                     [2F00]
2160 wp=x:sp=y:GOSUB 7050:POKE h!+a,32:L
     OCATE x,y:PRINT" ";:RETURN             [A7B8]
2170 '                                      [9722]
2180 '==============================        [717C]
2190 '* Eigenes Schiff getroffen *          [EA74]
2200 '==============================        [DB6E]
2210 '                                      [BD18]
2220 BORDER 26,0:INK 0,26,0:FOR i=1 TO 2
     0:NEXT:b=a:GOSUB 2380:GOSUB 7230:a=
     b:GOSUB 7040:INK 0,0:BORDER 0:b2=0:
     RETURN                                 [230A]
2230 '                                      [9B1C]
2240 '===========================           [5308]
2250 '* Energiefeld getroffen *             [6DBC]
2260 '===========================           [B30C]
2270 '                                      [9724]
2280 POKE h!+a,32:q=q+1:GOSUB 4040:INK 3
     ,26,6:LOCATE x,y:PRINT" ";:GOSUB 23
     80                                     [BFB2]
2290 wp=x:sp=y:GOSUB 7050                   [3604]
2300 POKE a+h!,32:b1=b1+1:f1=f1+1:POKE f
     p!+c,PEEK(fp!+(f-1)*2):POKE fp!+c+1
     ,PEEK(fp!+(f-1)*2+1):f=f-1:GOSUB 51
     20                                     [5FAA]
2310 INK 3,8:RETURN                         [E932]
2320 '                                      [C21C]
2330 '==========================            [EE8E]
2340 '*** Schiff getroffen ***              [95FE]
2350 '==========================            [2092]
2360 '                                      [E624]
2370 GOSUB 2290                             [96B2]
2380 GOSUB 4040:FOR i=0 TO 5:PEN INT(RND
     *3+1):LOCATE wp,sp:PRINT CHR$(244+i
     ):FOR j=1 TO 50:NEXT j,i:POKE h!+a,
     32:RETURN                              [DA96]
2390 '                                      [992A]
3000 '=============                         [7D46]
3010 '* Spielende *                         [EF70]
3020 '=============                         [954A]
3030 '                                      [BA1A]
3040 BORDER 0:INK 0,0:hi!=MAX(hi!,b1):PO
     KE fp!,hi!\256:POKE fp!+1,hi!MOD 25
     6:GOSUB 5140                           [B2C0]
3050 FOR a=1 TO 10:GOSUB 4040:FOR b=1 TO
      80:NEXT b,a                           [7F10]
3060 CLS                                    [2196]
3070 PEN 3:LOCATE 6,6                       [DC94]
3080 IF r1<0 THEN PRINT"PECH GEHABT !":P
     EN 2:PRINT," Sie haben es nicht ges
     chafft,"," die Uebermacht war zu gr
     oss !"ELSE PRINT"GERETTET !":PEN 2:
     PRINT," Sie koennen in Ruhe Ihr Tri
     ebwerk"," reparieren und diesen unf
     reundlichen<4>Raumsektor verlassen
     !"                                     [3362]
3090 PEN 1:LOCATE 6,14:PRINT"Noch ein Sp
     iel (J/N) ? ";                         [69EE]
3100 j$="":WHILE j$<>"J"AND j$<>"N":j$=U
     PPER$(INKEY$):WEND                     [54F6]
3110 CLS:IF j$="J"THEN RUN 540 ELSE END     [278C]
3120 '                                      [931A]
4000 '=============                         [15CE]
4010 '*>  Toene <*                          [DF6A]
4020 '=============                         [5BD2]
4030 '                                      [9D1C]
4040 CALL &BCA7:ENV 1,3,5,3,1,0,10,2,-3,
     15,9,-1,25                             [BF3C]
4050 SOUND 7,250,0,0,1,0,6                  [6BBE]
4060 RETURN                                 [9594]
4070 '                                      [9924]
```

Listing 6. Die einzelnen Unterprogramme zur Behandlung von Kollisionen etc.

Programmiersprachen für Einsteiger und Umsteiger

Die Vielfalt der Programmiersprachen wird immer größer und der Einsteiger oder Umsteiger, der seinen Erfahrungshorizont erweitern und statt Locomotive-Basic einmal etwas anderes ausprobieren möchte, steht hilflos vis-à-vis.

In den letzten Jahren hat die Entwicklung auf dem Gebiet der Computer-Hardware große Fortschritte gemacht. Die technischen Fähigkeiten moderner Computersysteme eröffnen dem Benutzer ein weites Feld an Anwendungsgebieten. Doch die Entwicklung auf der technischen Seite ist nichts wert, wenn sie nicht durch die entsprechende Software ausgenutzt wird.

Programmiersprachen haben im Computergewerbe einen hohen Stellenwert, weil sie die Werkzeuge zum Entwickeln von Software darstellen. Nur mit leistungsfähigen Programmiersprachen ist es möglich, innerhalb einer vernünftigen Zeitspanne anspruchsvolle Software zu entwickeln.

Da sich in letzter Zeit mehr und mehr Programmiersprachen auf dem Software-Markt tummeln, wollen wir im folgenden Beitrag dem interessierten CPC-Besitzer die Sprachen Basic, Pascal, Comal und Logo vorstellen, die dem Einsteiger und Umsteiger den Zugang zur Welt der Computer erleichtern.

Grundsätzlich ist zu sagen, daß es zwei Arten von Sprachen gibt: Interpreter und Compiler. Der Interpreter ist ein Systemprogramm, das ein in einer höheren Programmiersprache geschriebenes Programm beim Bearbeiten Befehl für Befehl übersetzt und direkt ausführt. Weil jedoch beim Ausführen eines Befehls neben der Übersetzung und den gesamten Fehlerabfang-Routinen auch noch die vom jeweiligen Befehl benötigten Systemroutinen aufgerufen werden müssen, ist die erreichbare Verarbeitungsgeschwindigkeit in der Regel gering.

Ein Compiler übersetzt hingegen den vorliegenden Quelltext nach erfolgreicher Fehlerbeseitigung komplett in einen maschinennahen Code, der natürlich wesentlich schneller abgearbeitet werden kann.

Um Ihnen die in diesem Beitrag vorgestellten Programmiersprachen anschaulicher darstellen zu können, geben wir zu jeder Sprache ein Beispielprogramm. Alle Programme führen eine Schleife mit bedingter Verzweigung sowie einer arithmetischen und einer logischen Rechnung aus. Die besondere Struktur der jeweiligen Programmiersprache wird in den unterschiedlichen Listings deutlich. Zudem geben die Beispielprogramme als Benchmarktests Aufschluß über die erzielte Geschwindigkeit (Bild).

Es ist jedoch zu beachten, daß ein Vergleich der Ablaufgeschwindigkeiten die Leistungsfähigkeit der Programmiersprachen nur unzureichend wiedergibt. Doch werden die Vor- und Nachteile der jeweiligen Sprachen im folgenden ausführlich hervorgehoben.

Die bekannteste und am weitesten verbreitete Programmiersprache ist Basic (Abkürzung für »beginners all purpose symbolic instruction code«). Fast jeder Heimcomputer hat einen Basic-Interpreter fest eingebaut, der nach dem Einschalten des Gerätes automatisch aktiviert wird.

So verfügt auch der Schneider CPC über einen integrierten Basic-Interpeter – das Locomotive-Basic.

Leider gibt es von Basic, das bereits Anfang der sechziger Jahre am College von Dartmouth entwickelt wurde, viele Dialekte, so daß die meisten Programme aufgrund der speziellen Befehlserweiterungen nicht auf andere Computer übertragbar sind.

Als Einsteigersprache für einen Computerbesitzer ist Basic jedoch ideal. Klar verständliche Schlüsselworte und Befehlsfolgen sowie die schnell erlernbare Befehlssyntax erleichtern Anfängern die Beschäftigung mit dieser Programmiersprache.

In letzter Zeit wird Basic zwar von vielen Programmierern etwas belächelt, doch völlig zu Unrecht. Denn aufgrund ständiger Weiterentwicklungen (zum Beispiel die Verbesserung des strukturellen Aufbaus) erfreut sich die Sprache weiterhin zunehmender Beliebtheit. Man denke nur daran, welches Aufsehen Turbo-Basic kürzlich im PC-Bereich erregte.

Daß sich moderne Basic-Versionen auch für professionelle Programmierung eignen, beweisen viele kommerzielle Programme, die ursprünglich in Basic geschrieben und dann mit einem Compiler übersetzt wurden.

Zwei interessante Versionen für die Computer der CPC-Reihe sind der CBasic-Compiler von Digital Research und natürlich das Locomotive-Basic.

Basic – nicht nur für Anfänger

Das Locomotive-Basic ist im Vergleich zu anderen Basic-Dialekten (Interpretern) sehr schnell, weil es einen Zwischencode (durch einen Compreter) erzeugt und einen relativ großen Sprachumfang besitzt, der viele Besonderheiten (zum Beispiel die Unterstützung von Interrupt-gesteuerten Unterprogrammen) aufweist.

Die Schwachpunkte liegen vor allem in einer Inkompatibilität, die zwischen den verschiedenen CPC-Versionen besteht, dem Fehlen wichtiger Grafikroutinen (beispielsweise eine Kreis- oder Rechteckfunktion) und der zu geringen Unterstützung für die Entwicklung strukturierter Programme.

Der Editor von Locomotive-Basic arbeitet zeilenorientiert. Jeder Programmzeile muß eine Zeilennummer vorangestellt sein, deren Betrag die relative Position der Zeile innerhalb eines Basic-Programms repräsentiert.

Im Gegensatz zum Locomotive-Basic stellt CBasic eine Programmiersprache dar, die auf dem CPC unter der Kontrolle des Betriebssystems CP/M 2.2 beziehungsweise CP/M Plus läuft. CBasic ist also nur für den Anwender von Interesse, der mindestens ein Diskettenlaufwerk (besser zwei) besitzt.

CBasic ist ideal vor allem für die Benutzer, denen die Geschwindigkeit des Basic-Interpreters nicht ausreicht. Der Compiler von Digital Research eignet sich – zu einem Preis von 174 Mark – auch zum Schreiben kommerzieller Software in fast allen Geschäftsbereichen. Dabei spielt es eine wichtige Rolle, daß der Compiler Maschinencode erzeugt und die Programmierung und Ausführung einzelner Programmteile erlaubt, die dann später zusammen das komplette Programm ergeben.

Der umfangreiche Befehlssatz ermöglicht eine komfortable Bedienung und Programmierung. Ein weiterer Vorteil von CBasic ist leider nur für CP/M-Plus-Besitzer von Interesse: Der CBasic-Compiler enthält nämlich integrierte Grafikbefehle und Funktionen für eine sehr komfortable Grafikprogrammierung.

Diese Grafikerweiterung kann aber nur dann aktiviert werden, wenn ein GSX-Treiber zur Verfügung steht. Da die Kommandos völlig unabhängig von der verwendeten Peripherie sind, erlaubt der CBasic-Compiler die Ausgabe von Grafiken auf jedem beliebigen angeschlossenen Ausgabegerät (Plotter, Drucker oder Bildschirm) ohne Änderung oder erneute Compilierung des Programms.

Ein weiteres Plus ist die hohe Rechengenauigkeit. Die verwendete vierzehnstellige Dezimal-Arithmetik garantiert Ergebnisse, die nicht durch Rundungsfehler beeinflußt sind, wie sie bei der oft eingesetzten Binärarithmetik auftreten können. Da der Compiler auch Integer-Arithmetik unterstützt (das heißt Rechnen mit ganzzahligen Werten), kann der Geschwindigkeitsgewinn bei compilierten Programmen noch wesentlich höher ausfallen.

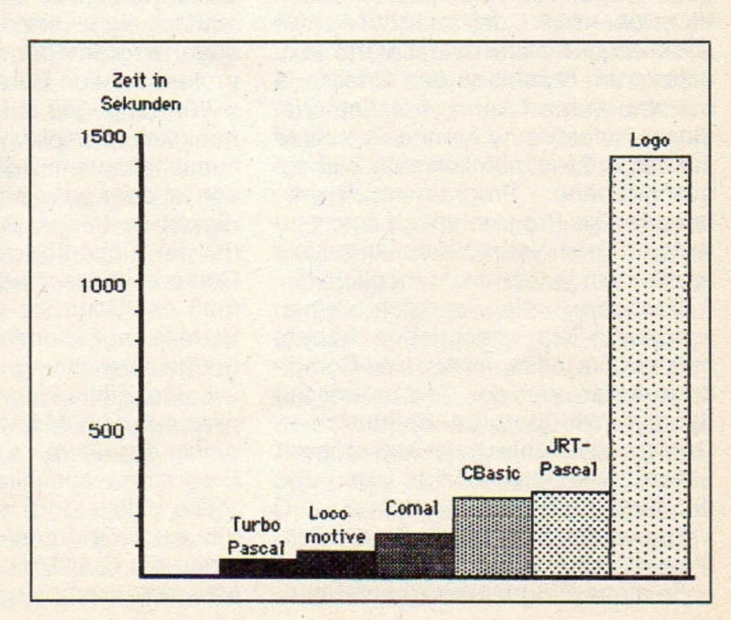
Wenn der Compiler jedoch mit Fließkommazahlen arbeiten muß, ist er auffallend langsam, weil die hohe Rechengenauigkeit ihren Tribut fordert. Selbst der Locomotive-Basic-Interpreter ist hier schneller (siehe Benchmarktest). Seine Geschwindigkeitsvorteile kann CBasic demnach nur im Integer-Bereich ausspielen.

CBasic für Mathematiker

Da der Compiler über die Möglichkeit verfügt, mehrzeilige Funktionen zu unterstützen, beinhaltet er Fähigkeiten, die sonst nur von anderen Programmiersprachen wie zum Beispiel Pascal bekannt sind. Innerhalb dieser Funktionen beziehungsweise Unterprogramme können lokale Variablen definiert werden, die nur in der angesprochenen Routine Verwendung finden.

Vom CBasic-Compiler existieren neben der Z80-Version auch Ausführungen für andere 8-Bit- und 16-Bit-Computer. Aus diesem Grund ist eine Portabilität auf ein anderes Betriebssystem (zum Beispiel MS-DOS oder DOS Plus) kein großes Problem, denn die Kompatibilität zwischen den einzelnen Versionen ist gewährleistet, so daß auf ein großes Potential an Quellcode-Programmen zurückgegriffen werden kann.

Die Geschwindigkeit der Programmiersprachen im Vergleich. Turbo-Pascal ist klarer Testsieger, doch auch das Locomotive-Basic schlägt sich beachtlich.



Das umfangreiche Handbuch zu CBasic ist in englischer Sprache geschrieben. Für Anfänger, die keine Erfahrung im Umgang mit englischen Texten besitzen, ist die Dokumentation nicht geeignet. Da sie jedoch sehr übersichtlich aufgebaut und in verständlichem Englisch verfaßt ist, sollten keine Probleme bei Anwendern auftreten, die grundlegende Englischkenntnisse besitzen.

Linker und Library

Manche Compiler übersetzen den Quellcode eines Programms in eine Art Zwischencode (sogenannter P-Code), der zwar schneller als ein interpretiertes Programm ist, aber dennoch nicht das Optimum an Geschwindigkeit darstellt. Der CBasic-Compiler hingegen benutzt einen Programmbinder (Linker) und eine Bibliothek (Library) im Zusammenhang mit dem eigentlichen Compiler, um reinen Maschinencode zu erzeugen, den der Computer direkt ausführen kann.

Der Compiler übersetzt in Zusammenarbeit mit dem Linker und der Library den Quellcode in ein Programm, das den zur Verfügung stehenden Speicherplatz effizient ausnutzt. Durch dieses Prinzip erhalten Sie die Möglichkeit, die einzelnen Bestandteile eines Programms unabhängig voneinander zu entwickeln, um sie dann nacheinander zu einem Komplex zusammenzufügen. Dieser modulare Aufbau erleichtert die Programmentwicklung ungemein.

Die Struktur von CBasic bedingt aber auch einige Nachteile. Der Quellcode eines Programms muß immer mit einem Editor verfaßt werden, der unabhängig vom Compiler arbeitet. Es ist lobenswert, daß auch der auf der CP/M-Plus-Diskette mitgelieferte Texteditor zum Schreiben des Quellcode benutzt werden kann, trotzdem hat dieser Aufbau eine Menge Probleme zur Folge. Es ist nämlich sehr wichtig, geschriebene Programme beziehungsweise Programmteile sofort zu testen. Dieses ist mit dem vorgestellten System jedoch nicht möglich.

Nachdem Sie nämlich einen Programm-Text geschrieben haben, müssen Sie jedesmal erst den Compiler aufrufen, der den Text zuerst auf Syntaxfehler überprüft. Sollte sich im Quelltext ein Fehler befinden, müssen Sie erst wieder den Editor laden und den Fehler beseitigen.

Eine weitere Schwäche ist die Zeit, die benötigt wird, um den Maschinencode eines Programms zu erzeugen. Der CBasic-Compiler besteht – wie bereits erwähnt – aus drei Teilen: erstens der Compiler, der CBasic-Programme in relozierbare Maschinencode-Module übersetzt, zweitens die Library, die alle Arithmetik- sowie Ein- und Ausgaberoutinen enthält, und drittens der Linker, der den compilierten Code mit den Library-Routinen zu einem lauffähigen Programm kombiniert, das sich vom Computer direkt ausführen läßt.

Aus diesem Grund müssen alle Programme zusätzlich zum Schreiben des Quellcodes noch die Compilierungs-, Library- und Linkphase durchlaufen. Das dauert natürlich ziemlich lange, weshalb schon einige Erfahrung im Umgang mit Basic-Programmen und ihren Fehlern nötig ist, um nicht zu verzweifeln.

Die beiden besprochenen Basic-Versionen sprechen recht unterschiedliche Zielgruppen an. Während sich das Locomotive-Basic auch für »blutige« Anfänger eignet, die noch keinerlei Erfahrung mit Computern vorweisen können, wendet sich CBasic vor allem an Anwender, die die Grundzüge der Basic-Programmierung beherrschen und mit der erreichten Geschwindigkeit des ohnehin schon recht schnellen eingebauten Basic-Interpreters nicht mehr zufrieden sind.

Interpreter oder Compiler?

Sicherlich gehört Locomotive-Basic zu den besten Basic-Dialekten, die der Heimcomputermarkt zu bieten hat. Die Unterstützung von Fenstern und das ausgetüftelte Interrupt-Konzept sprechen für sich. Da die erreichbare Geschwindigkeit auch zufriedenstellend ist, eignet sich Locomotive-Basic sogar begrenzt für Anwendungen im professionellen Bereich.

Wer hingegen auf hohe Rechengenauigkeit, beispielsweise in den wissenschaftlichen Bereichen angewiesen ist oder auf eine hohe Geschwindigkeit im Integer-Bereich Wert legt, der ist sicherlich mit dem CBasic-Compiler besser beraten. Allerdings muß der Benutzer zugunsten dieser Vorteile auf Komfort bei der Programmentwicklung verzichten.

Leider gibt es zur Zeit keinen Compiler auf dem Markt, mit dem Basic-Anwender Ihre Locomotive-Basic-Programme compilieren könnten.

Die existierenden Compiler arbeiten nur mit Integer-Variablen, was ihnen die Qualifikation für viele interessante Anwendungen abspricht. Sie sind auch nicht voll kompatibel zum Locomotive-Interpreter, so daß viele Befehle nicht umgesetzt werden können.

Eine weitere Programmiersprache, die nicht nur im Heimcomputerbereich weite Verbreitung erlangte, ist Pascal.

Pascal ist eine universell einsetzbare, höhere Programmiersprache, die vom Schweizer Professor Niklaus Wirth an der Technischen Hochschule in Zürich entwickelt wurde und ihren Namen zu Ehren des Physikers Blaise Pascal erhielt.

Pascal – strukturiert programmiert

Die im Jahre 1971 veröffentlichte exakte Definition der Programmiersprache Pascal erlaubte dem Benutzer erstmals eine systematische Annäherung an die Computer-Programmierung. Der festgelegte Sprachschatz von Pascal behandelt insbesondere die Unterstützung der strukturierten Programmierung.

Seit dieser Zeit wurde Pascal auf fast jedem Computer implementiert, um mit dieser Sprache nahezu jede Programmieraufgabe zu lösen.

Pascal entwickelte sich im Laufe der Jahre zu einer modernen Hochsprache, die sich bei Programmierern aller Sparten großer Beliebtheit erfreut. Dabei reicht das Anwendungsgebiet vom Einstieg in die Heimcomputerwelt bis hin zum Einsatz bei der professionellen Programmierung.

Turbo-Pascal aus dem Hause Borland stellt den am weitesten verbreiteten, leistungsfähigsten und mit 225 Mark auch preiswertesten professionellen Pascal-Compiler dar. Dieses Programmiersystem eignet sich einerseits hervorragend für den Einstieg in die strukturierte Programmierung und andererseits ist es ein herausragendes Produkt für den Programmentwickler, der eine effizient einsetzbare Programmierumgebung sucht.

Die extrem kurze Compilationszeit und die hohe Ausführungsgeschwindigkeit machen Turbo-Pascal zu einem äußerst leistungsfähigen Programmierwerkzeug. Allein der integrierte Wordstar-kompatible Editor wurde zu einem Meilenstein unter den Editoren für Programmiersprachen, an dem sich alle anderen Editoren messen lassen müssen.

Da der Turbo-Pascal-Editor bildschirmorientiert arbeitet, ist das Schreiben von Programmen mit ihm fast ein Kinderspiel.

Den Durchbruch erzielte Turbo-Pascal jedoch nicht nur wegen seiner

Geschwindigkeitsvorteile, sondern auch mit der kompakten Struktur des Programmiersystems. Alle Komponenten für die erfolgreiche Erzeugung des Maschinencodes befinden sich in einem Programm. Das heißt, daß Editor, Compiler, Linker und Library zusammen eine kompakte Datei bilden.

Eine völlig andere Systemstruktur besitzt das Public Domain-Produkt JRT-Pascal, das bereits für 30 Mark erhältlich ist.

Doch zuerst einige historische Details zu JRT-Pascal. Der Programmautor Jim Tyson bot diesen Pascal-Compiler im Jahre 1982 durch Anzeigen in amerikanischen Computerzeitschriften zum damals sagenhaften Preis von 29,95 US-Dollar an. Was sich dann ereignete, klingt wie ein Märchen. Tysons Firma JRT-Systems erhielt in kürzester Zeit über einhunderttausend Bestellungen. Als Einmannbetrieb war JRT-Systems natürlich völlig überfordert, alle Bestellungen zu bearbeiten. Tyson bekam deshalb Ärger mit der amerikanischen Bundeshandelskommission, weil er die im voraus bezahlte Ware nicht liefern konnte.

So mußte Tyson seine Firma schließen, und er gab den Compiler zur Weiterverbreitung als Public Domain-Programm frei. Dadurch hat sein Pascal-Compiler bis heute viele Freunde in der ganzen Welt gefunden. (Eine verbesserte Version des JRT-Pascal-Compilers wird seit einigen Jahren von Ellis-Computing unter dem Namen Nevada-Pascal angeboten.)

Fast umsonst: JRT-Pascal

JRT-Pascal läuft auf dem CPC, ebenso wie Turbo-Pascal unter CP/M. Das heißt, daß für das Arbeiten mit JRT-Pascal mindestens ein Diskettenlaufwerk notwendig ist.

Allerdings benötigt JRT-Pascal eine Speicherkapazität von mindestens 128 KByte. Im Gegensatz zu Turbo-Pascal arbeitet JRT-Pascal nach einem System, wie es von vielen Programmiersprachen (zum Beispiel auch vom zuvor besprochenen CBasic) bekannt ist.

So wird der Quellcode eines Programms erst mit einem Texteditor geschrieben und darauf compiliert. Bis hier gleicht das Verfahren dem bei Turbo-Pascal. Logische Programmfehler, die erst bei Ablauf des Programms auftreten, können jedoch nur nach erneutem Aufruf des Editors beseitigt werden, während Turbo-Pascal automatisch den Editor aufruft und den Textcursor an der fehlerhaften Stelle positioniert.

JRT-Pascal lehnt sich eng an den Standard des Informatikprofessors Niklaus Wirth an. Darüber hinaus enthält es jedoch eine umfangreiche Befehlserweiterung, die eine komfortable Programmierung erlaubt.

So liegen die Stärken von JRT-Pascal eindeutig in der Stringverwaltung, die theoretisch Zeichenketten bis zu einer Länge von 64 KByte zuläßt. Mit Hilfe zahlreicher Befehle kann die Form einer Zeichenkette in vielerlei Hinsicht geändert werden.

Ein weiterer Vorteil von JRT-Pascal liegt im sogenannten Extern-System. Das Extern-System erlaubt die Entwicklung von Programmen, deren Länge weit über die Speicherkapazität des Computers hinausgeht. Im Gegensatz zu anderen Programmiersprachen, die nach diesem Prinzip arbeiten, verwaltet JRT-Pascal die benötigten Module beziehungsweise Funktionen in einer besonderen Art.

Benötigt das Hauptprogramm ein externes Programm, so wird die betreffende Datei bei Bedarf in den Speicher geladen und ausgeführt. Der von den Prozeduren belegte Speicherplatz wird erst dann wieder freigegeben, wenn er vom Hauptprogramm anderweitig benötigt wird. Interessant dabei ist, daß JRT-Pascal in diesem Fall nicht ein beliebiges Modul im Arbeitsspeicher löscht, sondern das am längsten nicht mehr benutzte!

Zusammenfassend kann man sagen, daß die beiden vorgestellten Pascal-Compiler zwei interessante Alternativen aus der Pascal-Welt darstellen. Dabei eignet sich die preiswerte Public Domain-Version eher zu ersten Einstiegsversuchen. Sollten Sie dann stärkeres Interesse an dieser Programmiersprache finden, so führt am Erwerb von Turbo-Pascal kein Weg vorbei.

Comal spielt auf dem Gebiet der Programmiersprachen noch eine untergeordnete Rolle. Neben Basic haben allenfalls Pascal (als Musterbeispiel für strukturiertes Programmieren) und C (als maschinennahe Hochsprache) einen hohen Bekanntheitsgrad erlangt.

Doch die geringe Verbreitung von Comal erstaunt, wenn man erfährt, daß diese Sprache die herausragenden Merkmale der anderen wichtigen Hochsprachen fast vollständig in sich vereint.

Comal – genauer Comal-80 – wurde im Jahr 1973 von den Dänen Christensen und Löfsted entwickelt, um eine Programmiersprache zu schaffen, die sich für den Einsatz in Schulen und Universitäten eignet. Das Fundament von Comal wird aus Sprachelementen von Basic und Pascal gebildet. Doch Comal wurde laufend aktualisiert, so daß sich in der vorliegenden Version auch Elemente von Logo und C finden lassen.

Mit Comal sollten viele umständliche Verfahren herkömmlicher Programmiersprachen vermieden, gleichzeitig aber eine akzeptable Ablaufgeschwindigkeit erreicht werden. So ist die Programmierumgebung zum Beispiel ein Interpreter, der den zeitaufwendigen Übersetzungsvorgang der Compilersprachen umgeht.

Zudem ist Comal so konzipiert, daß es nicht nötig ist, eine exakte Definition der Programmstruktur durchzuführen. Der Anwender muß beispielsweise die benötigten Variablen nicht, wie in Pascal üblich, zu Programmbeginn gesondert deklarieren, um sie im Hauptprogramm benutzen zu dürfen.

Comal – die unbekannte Sprache

Die Hauptforderungen und Ziele, denen Comal-80 gerecht werden will, lassen sich wie folgt zusammenfassen:

Der Programmierer kann Befehle, Prozeduren und Funktionen ähnlich wie in Forth (siehe Beitrag über Sprachen für Profi-Programmierer) neu definieren und diese im Direktmodus aufrufen.

Der Anwender kann alle Fehlerzustände, die durch Umsetzung eines Algorithmus auf den Computer erfolgen, behandeln und gezielt beseitigen.

Der strukturierte Aufbau eines Comal-Programms wird erreicht, indem die Sprache viele spezifische Funktionen des speziellen Computers unterstützt und durch ein sinnvolles Package-Konzept (ähnlich den Include-Dateien) Unübersichtlichkeit vermeidet.

Hinzu kommen ein benutzerfreundlicher Bildschirmeditor, klare Fehlermeldungen in Deutsch und die Übertragbarkeit einzelner Programme auf andere Computertypen. Comal ist nicht, wie oft vermutet wird, eine simple Basic-Erweiterung, sondern stellt dem Programmierer eine komfortable Programmierumgebung zur Verfügung. Die außerordentlich schnellen Diskettenoperationen, die Programmstrukturen und die umfassenden Arithmetikroutinen sind weitere Vorteile dieser Programmiersprache.

Im Gegensatz zur Comal-Version 1.83, die bislang für 69 Mark auf Diskette vertrieben wurde, ist nun für alle CPC-Modelle auch die Version 2.0 für stolze 248 Mark als ROM-Modul erhältlich.

Doch die Version 2.0 ist ihr Geld wert, denn sie enthält im Gegensatz zu der älteren Fassung einen wesentlich erweiterten Befehlssatz, der keine Wünsche mehr offenläßt. Sie unterstützt das 3-Zoll-Laufwerk von Schneider ebenso wie die Diskettenstation von Vortex unter VDOS 2.0.

Sollten in Ihrem Computer mehr als 64 KByte Speicherplatz zur Verfügung stehen, so wird dieser zusätzliche Bereich als RAM-Disk benutzt. Das funktioniert unter Comal sowohl mit den oberen 64 KByte des CPC 6128, als auch mit der Vortex-Speichererweiterung, den dk'Tronics-Produkten (inklusive Silicon Disc) und der Data Media-Aufrüstung!

Da macht das Programmieren Spaß

Für Comalprogramme und Datenfelder sind im Gegensatz zur Version 1.83 35 KByte Speicherplatz verfügbar. Das ROM-Modul enthält jedes Package auf EPROM; man muß sie also nicht erst von Diskette laden. Um dem Programmierer bei der Programmanalyse und Fehlerbeseitigung eine Hilfe zu geben, erscheint jede Fehlermeldung in Deutsch mit Kommentar.

Erfreulich ist auch die Fähigkeit der Comal-Version 2.0, relative Dateien einzurichten, die sehr leicht und gut kontrollierbar zu lesen und zu verändern sind.

Außerdem lassen sich mit dieser Version sequentielle Dateien im Append-Modus betreiben. Das heißt, bereits angelegte, beschriebene und wieder geschlossene Dateien dürfen zum Weiterbeschreiben (Anhängen) geöffnet werden.

Doch einen wichtigen Nachteil von Comal wollen wir nicht verschweigen. Das CPC-Comal bearbeitet Funktionen wie Sinus, Kosinus oder Logarithmus äußerst langsam.

Für den CPC 6128 mit dem Betriebssystem CP/M Plus gibt es eine weitere Comal-80-Version. Sie wird zusammen mit einem Runtime-Modul ausgeliefert. Dadurch kann man in Comal geschriebene Programme auch ohne den eigentlichen Interpreter unter der Programmierumgebung CP/M ablaufen lassen.

Das Comal-ROM-Modul begleitet ein ausführliches deutsches Handbuch mit vielen Beispielprogrammen.

Zusammenfassend stellt die Comal-Version 2.0 sowohl für den Einsteiger als auch für den Profi eine interessante Programmiersprache dar, die die Entwicklungszeit von Software erheblich verkürzt (wenn auch zu einem stolzen Preis).

Die letzte Programmiersprache, die wir in diesem Beitrag vorstellen, ist in der Programmiererszene sehr umstritten. Es handelt sich um Logo. Ohne Frage ist Logo die Programmiersprache, die bereits Kindern im Vorschulalter spielerisch den Einstieg in die Computerwelt erleichtert.

Doch lassen sich mit Logo durchaus auch anspruchsvolle Programme schreiben. Auf dem Gebiet der Grafikprogrammierung ist diese Sprache ohnehin fast unschlagbar. Das Prinzip der Turtle-Grafik findet neuerdings auch in anderen Software-Produkten großen Anklang.

Entwickelt wurde Logo in langjähriger Arbeit von Seymore Papert. Es entspricht in seinen Grundstrukturen den Definitionen von Lisp. Es ist gleichzeitig aber übersichtlich aufgebaut und leicht verständlich, da sich der Programmautor lange mit kindlichen Verhaltensweisen und Denkstrukturen beschäftigt hat. Von Pädagogen empfohlen, gilt Logo als eine der besten Einsteigersprachen überhaupt.

Logo – keine Spielerei

Der Aufbau von Logo weicht von dem einer »herkömmlichen« Programmiersprache ab. Die einzelnen Befehle von Logo werden einfach eingetippt, wie auch ein normaler Text geschrieben wird. Existiert ein Befehl oder ein Routinenaufruf nicht, so kann er anhand der übrigen Befehle selbst definiert werden.

Der »Preis« ist natürlich eine der stärksten Seiten von Logo, denn die Version Dr. Logo liegt jedem Schneider-Diskettenlaufwerk als »kostenlose Zugabe« bei.

Wer jedoch professionell programmieren möchte, sollte die Finger von Logo lassen. Die hohe Anwenderfreundlichkeit und der große Programmierungskomfort werden nämlich durch einige schwerwiegende Nachteile erkauft. So benötigen die leicht verständlichen Logo-Routinen soviel Speicherplatz, so daß umfangreiche Programme von vornherein nicht in Frage kommen.

Das gravierendste Problem liegt jedoch zweifelsohne in der geringen Ablaufgeschwindigkeit von Logo (siehe Benchmarktest), die bei rechenintensiven Problemlösungen besonders ins Gewicht fällt. Aus diesem Grund ist Logo für mathematische Aufgaben denkbar ungeeignet.

In den letzten Jahren sind immer leistungsfähigere Versionen der Programmiersprache Basic entwickelt worden, doch die Grenzen sind abzusehen. So ist Basic sicherlich die am wenigsten spezialisierte Sprache der vorgestellten Programmiersprachen, so daß es in fast allen Bereichen zum Einsatz kommt. Doch für Programme, die eine hohe Geschwindigkeit erreichen müssen, ist Basic – zumindest in der Interpreterform – ungeeignet.

Welche Sprache ist die richtige

Weil jedoch eine große Anzahl der Programme in Computerzeitschriften in dieser Sprache abgedruckt sind, kann man auch auf ein großes Potential interessanter und leicht umsetzbarer Software zurückgreifen, und das fast zum Nulltarif.

Der Programmiersprache Pascal gehört zumindest die nahe Zukunft. Vor allem das leistungsfähige Konzept von Turbo-Pascal und der Preis des Public Domain-Programms JRT-Pascal tragen zu der großen Popularität bei.

Turbo-Pascal setzte mit der herausragenden Geschwindigkeit der Compilierung und des Programmcodes, sowie mit der Kürze des Programmcodes bereits einen Meilenstein in der Entwicklung der Programmiersprachen.

Die exakte Definition der Programmeigenschaften durch Professor Nikolaus Wirth und der modulare Aufbau von Pascal bewirken die unkomplizierte Übertragbarkeit von einzelnen Programme auf andere Computertypen.

Die spezielle Anpassung der Grafikbefehle an die spezifischen Eigenarten der CPC-Grafik und der akzeptable Preis erleichtern dem interessierten Anwender den Einstieg in das interessante Gebiet der Pascal-Programmierung.

Wenn Sie erst einmal in Pascal hineinschnuppern möchten, dann empfiehlt sich die Public-Domain-Version. Sie enthält zu einem wirklich herausragenden Preis eine komplette Anleitung und ein sehr leistungsfähiges Programm.

Comal ist sicherlich eine der bisher am wenigsten beachteten Sprachen, die aber wesentlich mehr Interesse verdient. Die Nähe zu Basic, die den

Umstieg problemlos werden läßt und die Implementation der wichtigsten Eigenschaften anderer höherer Programmiersprachen machen Comal zu einer leistungsfähigen Programmierumgebung.

Da es eine Disketten- und eine Modulversion gibt, ist auch der Preis kein großes Problem. Die Modulversion hat natürlich den Vorteil, daß ein erheblich größerer Befehlssatz und wesentlich mehr Speicherplatz zur Verfügung stehen.

Der durchdachte Aufbau und ein komfortabler Editor machen Comal zu einer interessanten Programmiersprache, deren Entwicklung in bezug auf Ausführungsgeschwindigkeit und Bekanntheitsgrad für die Zukunft hoffen läßt.

Bücher, Bücher, Bücher

Abschließend möchten wir Sie noch auf einen nicht zu vernachlässigenden Umstand aufmerksam machen, der ebenfalls einen Einfluß auf Ihre Kaufentscheidung haben sollte.

Alle vorgestellten Programmiersprachen mit Ausnahme des Locomotive-Basic enthalten im Lieferumfang keine Anleitung zum Erlernen der Programmiersprache. Das heißt, daß Sie zu den Anschaffungskosten der Programmiersprache noch mit Kosten für begleitende Literatur rechnen müssen, die Ihnen den Umstieg von Basic auf eine andere Programmiersprache erleichtert.

Hier gibt es für einige Compiler eine größere Auswahl an Büchern als für andere, und der Verbreitungsgrad an Fachliteratur kann auch regional sehr unterschiedlich sein.

Es ist nicht verkehrt, wenn Sie sich vor dem Kauf mit Literatur zu einer Programmiersprache beschäftigen, um dann eine Kaufentscheidung zu treffen. So können Sie sich manche Enttäuschung ersparen.

Sie erhalten durch Bücher einen größeren Überblick und unter Umständen auch neue Anregungen, die wir Ihnen innerhalb dieses begrenzten Rahmens nicht geben konnten.

Wer weiß – vielleicht schreiben Sie später selbst einmal die Programmiersprache der Zukunft!

(Markus Zietlow/ma)

Bezugsquellen:
CBasic: Markt&Technik Verlag, Hans-Pinsel-Str. 2, 8013 Haar
Turbo-Pascal: Heimsoeth Software, Fraunhoferstr. 13, 8000 München 5
JRT-Pascsal: Martin Kotulla, Grabbestr. 9, 8500 Nürnberg 90
Comal: Bauer & Kaup GBR, Teichstr. 9, 4401 Saerbeck

```
10 PRINT"Start"
20 i=0
30 WHILE i<>5000
40 i=i+1
50 IF i/2=INT(i/2) THEN i=i*2/2+2-2
60 a=(5 AND 4 OR 3)
70 WEND
80 PRINT"Ende"
```

Listing 1. Der Benchmarktest im Locomotive-Basic

```
PRINT "Start";chr$(7)
i=0
WHILE i<>5000
  i=i+1
  IF i/2=INT(i/2) THEN i=i*2/2+2-2
  a=(5 AND 4 OR 3)
WEND
PRINT chr$(7);"Ende"
END
```

Listing 2. Der CBasic-Compiler erlaubt eine bessere optische Strukturierung und benötigt keine Zeilennumerierung

```
program benchmark;

  var
    a,i:real;

  begin
    writeln('Start');
    i:=0;
    repeat
      i:=i+1;
      if i/2=int(i/2) then i:=i*2/2+2-2;
      a:=(5 and 4 or 3);
    until i=5000;
    writeln('Ende');
  end.
```

Listing 3. Die Turbo-Pascal-Fassung des Benchmarktests

```
program benchmark;

  var
    a,i:real;

  begin
    writeln('Start');
    i:=0;
    repeat
      i:=i+1;
      if round(i/2)=trunc(i/2) then i:=i*2/2+2-2;
      a:=5+4-3;
    until i=5000.0;
    writeln('Ende');
  end.
```

Listing 4. Für den JRT-Pascal-Compiler waren einige kleine Änderungen nötig (logische Operatoren werden nicht akzeptiert)

```
0010 PRINT "Start"
0020 i:=0
0030 REPEAT
0040 i:=i+1
0050 IF i/2=INT(i/2) THEN i=i*2/2+2-2
0060 a:=(5 AND 4 OR 3)
0070 UNTIL i=5000
0080 PRINT "Ende"
```

Listing 5. Der Benchmarktest als Comal-Programm ...

```
to benchmark
type "Start
make "i 0
repeat 5000 [make "i :i + 1 if (:i / 2 = int :i / 2)
  [make "i:i * 2 / 2 + 2 - 2]
  make "a 5 + 4 - 3]
type "Ende
end
```

Listing 6. ... und als Primitive in Logo (auch hier sind keine logischen Operatoren zugelassen)

Programmieren wie die Profis

Programmiersprachen gibt es wie Sand am Meer. Doch welche ist die richtige für eine bestimmte Anwendung? Wir haben mehrere interessante, aber noch wenig verbreitete Sprachen für den CPC auf ihre Leistungsfähigkeit und das Einsatzgebiet überprüft.

In den letzten zehn Jahren ist viel auf dem Gebiet der elektronischen Datenverarbeitung passiert. Fortgeschrittene Miniaturisierung und ein enormer Preisverfall machten Computer einem großen Interessentenkreis zugänglich. Doch das steigende Angebot an Hardware hat auch einen hohen Bedarf an Software zur Folge. Die Ausweitung der Anwendungsmöglichkeiten auf kaufmännische und wissenschaftliche Gebiete bedeutet auch eine erhöhte Nachfrage nach Programmiersprachen, die den gestellten Anforderungen genügen.

Prinzipiell lassen sich heutzutage zwei Trends beobachten, die die Entwicklung von Programmiersystemen nachhaltig beeinflussen. Als erstes besteht ein Bedarf an schnellen Sprachen zur Auswertung und Verwaltung von großen Datenmengen, zum Beispiel in der Forschung. Andererseits haben sich vor allem in letzter Zeit Programmiersprachen etabliert, die ein relativ beschränktes Zielgebiet zulassen. Diese problemorientierten Sprachen haben den Vorteil, daß sie sich jeweils für ein spezielles Aufgabenfeld am besten eignen. Die hautsächlichen Anwendungsgebiete der Sprachen sind in Bild 1 grafisch dargestellt. Bild 2 zeigt die Distanz zwischen der Programmiersprache und der Maschinenebene auf.

Fast alle Programmierumgebungen haben eins gemeinsam: Sie sind Compiler-orientiert. Das bedeutet, daß das Quellprogramm erst mit einem Texteditor geschrieben wird, um es danach in einen maschinennahen Code umzuwandeln. Grundsätzlich gibt es dabei zwei Alternativen. Einerseits besteht die Möglichkeit, ein Komplettsystem anzubieten, das schnell zwischen Editor und Compiler wechseln kann. Dies hat den Vorteil, daß das Schreiben und Überprüfen von Programmen äußerst komfortabel ist.

Das entgegengesetzte Prinzip geht davon aus, Editor, Compiler, Linker und Library zu trennen. Das hat den Vorteil, daß man nicht auf ein System beschränkt ist, sondern mit den Hilfsmitteln unterschiedlicher Compiler, Linker und Libraries arbeiten kann. Das offene System erlaubt auch die Verwendung mehrerer Programmteile, die in unterschiedlichen Sprachen geschrieben wurden.

Mit Hilfe dieses Artikels wollen wir Ihnen das Basiswissen über Programmiersprachen für fortgeschrittene Benutzer vermitteln und Ihnen die verschiedenen Versionen vorstellen, die auf den Schneider CPCs eingesetzt werden. Es ist wichtig anzumerken, daß die vorgestellte Auswahl weder vollständig ist noch sein kann. Dafür gibt es einfach zu viele exotische und problemspezifische Sprachen, die für den normalen Anwender uninteressant sind. Andererseits gibt es sehr viele verschiedene Versionen der einzelnen Sprachen, deren Aufzählung und Beschreibung eine ganze Bücherei füllen würde. So haben wir uns entschieden, Ihnen nur die wichtigsten und am weitesten verbreiteten Dialekte vorzustellen.

Der Nachfolger von Pascal

Die wahrscheinlich bekannteste höhere Programmiersprache ist Pascal. Sie ist das erste populäre Programmiersystem, das sich mit strukturierter Programmierung und gezielt definierten Datenstrukturen beschäftigt. Doch soll im folgenden keine Beschreibung von Pascal folgen, da deren Aufbau schon im ersten Beitrag über Programmiersprachen dargestellt wurde (siehe Programmiersprachen für Einsteiger). Pascal spielt aber bei der Programmierung auf PCs eine sehr wichtige Rolle, weshalb es hier nochmals erwähnt werden muß.

Mit Modula stellt sich eine Pascal sehr ähnliche Programmiersprache vor. Modula ist die konsequente Weiterentwicklung von Pascal, das in einigen Bereichen Schwächen aufweist. So unterstützt Pascal, das ursprünglich nur zu Lehrzwecken konzipiert wurde, keine Routinen, die das Betriebssystem direkt ansprechen oder auf bestimmte Speicherbereiche zugreifen. Des weiteren sind die Ein- und Ausgaberoutinen ziemlich unzureichend.

Im Jahre 1975 hat Niklaus Wirth wegen dieser Schwächen des Pascal-Systems eine neue Sprache entwickelt, die die auftretenden Schwachpunkte von Pascal vermei-

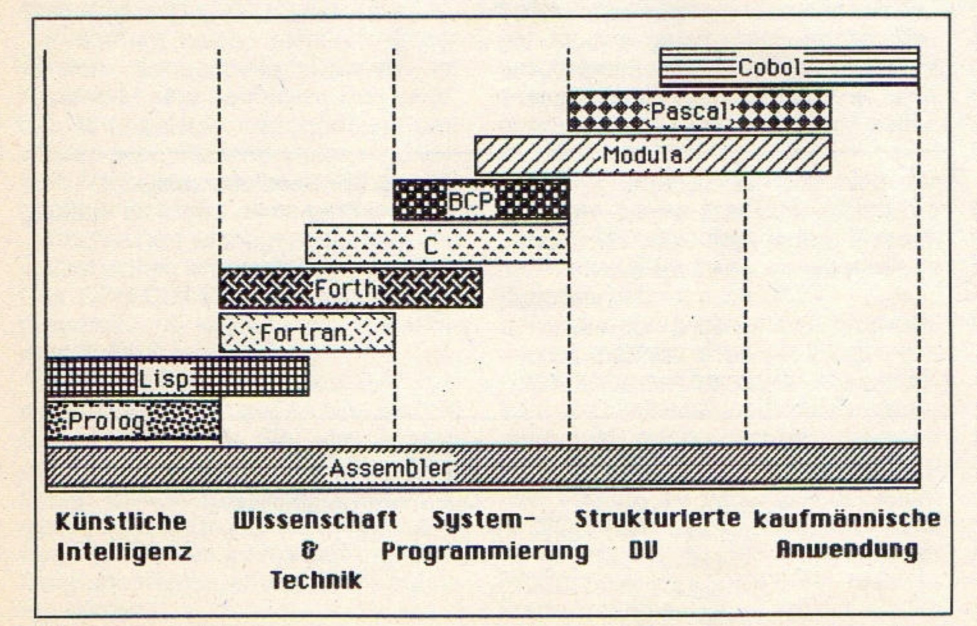


Bild 1. Die ursprünglichen Anwendungsgebiete der bekanntesten Programmiersprachen

det. Die Firma Borland, von der der Standard Turbo-Pascal stammt, hat nun vor einiger Zeit einen Modula-2 Compiler veröffentlicht, der unter CP/M lauffähig ist. Er heißt - wie sollte es auch anders sein - Turbo-Modula und erfüllt natürlich den Wirthschen Standard. Die Programmierumgebung lehnt sich eng an das von Turbo-Pascal bekannte Erscheinungsbild. Der Editor ist wie gewohnt Wordstar-kompatibel, enthält aber im Gegensatz zu der Turbo-Pascal-Version einige Besonderheiten.

Da es mit dem Modula-Editor möglich ist, Texte zu schreiben, die über den zur Verfügung stehenden Speicherplatz hinausgehen, benutzt er natürlich oft das Diskettenlaufwerk. Dadurch erscheint der Editor zwar sehr langsam, doch läßt sich die fehlende Geschwindigkeit wegen der großen Vorteile dieser Programmkonzeption akzeptieren. Turbo-Modula ist ein Dialekt, der sich hauptsächlich für fortgeschrittene Programmierer eignet. Wer Pascal perfekt beherrscht, wir mit Turbo-Modula seine Freude haben. Mit 225 Mark ist Turbo-Modula jedoch nicht gerade billig.

Die nächste Programmiersprache, die wir Ihnen vorstellen wollen, ist C. C wurde im Jahre 1972 von Ken Thomas und Dennis Ritchie entwickelt. Zu dieser Zeit trat das Problem auf, das neu entwickelte Betriebssystem Unix auf andere Rechner umzusetzen. Um in bezug auf Unix eine möglichst große Portabilität zu erreichen, entwickelten die beiden Autoren die Programmiersprache C. Sie ist sehr maschinennah und erlaubt sehr kompakte und schnelle Programme.

Viele Befehle, die normalerweise nur in Assembler zur Verfügung stehen, sind auch in C integriert. Durch die weite Verbreitung ist auch eine hohe Portabilität auf alle wichtigen Computer gegeben. C enthält viele heute übliche Datenstrukturen und unterstützt strukturiertes Programmieren. Die Möglichkeit von rekursiven Funktionsaufrufen und der modulare Aufbau machen C zu einer Programmiersprache, die Weichen für die nahe Zukunft stellt.

Ein äußerst unangenehmer Umstand tritt im Zusammenhang mit C jedoch bei der Fehlererkennung und -beseitigung auf. Wegen der maschinennahen Konstruktion erkennt C keine Fehler, soweit sie nicht die Syntax einer Funktion betreffen. Ein weiterer Nachteil von C ist, daß es zu wenige Plausibilitätsprüfungen durchführt. Das hat zwar einerseits sehr kurze Programme zur Folge, aber andererseits führt diese Vorgehensweise zu schwer aufzufindenden Fehlern.

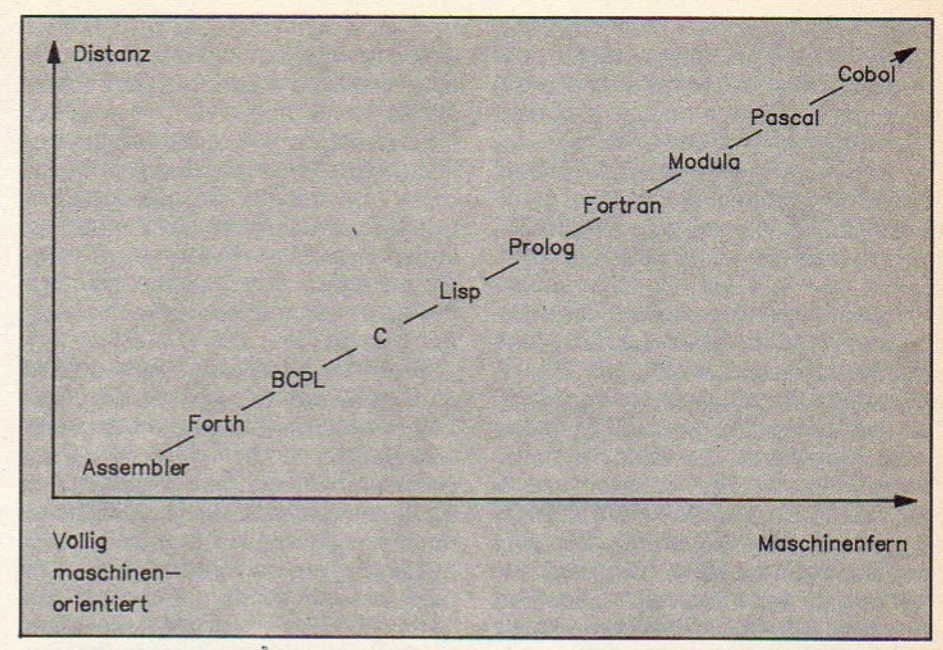


Bild 2. **Eine vereinfachte Darstellung des Verhältnisses zwischen Programmiersprache und Maschinennähe**

Für den Schneider CPC gibt es zwei Versionen, die wir besonders herausheben möchten. Sie tragen beide den gleichen Namen (Small C), sprechen aber recht unterschiedliche Anwenderkreise an. Die erste der beiden Versionen nennt sich Small-C-Entwicklungssystem und ist im Markt & Technik-Verlag erschienen. Sie ist wesentlich besser dazu geeignet, professionelle Programme zu entwickeln als die Public Domain-Version, benötigt aber für ihren Betrieb einen Speicherplatzumfang von mindestens 128 KByte RAM.

Ein Entwicklungssystem

Das Entwicklungssystem befindet sich auf drei beidseitig bespielten 3-Zoll-Disketten, die alle zum Betrieb des Compilers benötigten Unterroutinen enthalten. Ein sehr interessanter Punkt ist, daß alle Hauptprogramme als C-Quellcode geliefert werden und so später je nach Bedarf von erfahrenen Programmierern modifiziert werden können. Das Handbuch ist die schwächere Seite des Systems. Es befinden sich zwar Beschreibungen der Small-C-Bibliothek sowie des mitgelieferten Small-Mac-Assemblers und des Small-C-Tools darin, es fehlen aber Hinweise auf die Bedienung des umfangreichen Entwicklungspaketes. Diese C-Variante ist mit 99 Mark recht preiswert.

Die zweite Variante von Small C ist als Public Domain-Software für 30 Mark erhältlich. Dieser C-Compiler akzeptiert nur eine bestimmte Untermenge der C-Standardbefehle. Dabei wurde die Befehlsmenge so ausgewählt, daß die wichtigsten Prozeduren nur sehr gering beeinträchtigt sind.

Das gelieferte System besteht aus einem Texteditor, Assembler, Linker und natürlich dem eigentlichen Compiler. Wie beschrieben, ist die Fehlerkontrolle bei C sehr eingeschränkt. Um diese Fehlerquelle einzudämmen, enthält diese Version einen sogenannten Profiler, der die Fehleranalyse und -beseitigung wesentlich vereinfacht. Der Profiler speichert die Anzahl der einzelnen Funktionsaufrufe und gibt sie am Ende eines Programmes aus. Dadurch erhalten Sie die Möglichkeit, zu häufig aufgerufene Routinen in ihrer Arbeitsweise zu rationalisieren und andererseits bemerken Sie, welche von Ihnen definierten Funktionen nicht aktiviert wurden.

Hierzu ist anzumerken, daß die Dokumentation zu dem Programmpaket keine Einführung in die Programmiersprache darstellt. Sie ist lediglich eine Ausführung der Besonderheiten des Systems und enthält Hinweise zur Steuerung der Programmentwicklung. Erfreulicherweise ist sie komplett ins Deutsche übersetzt, was bei Public Domain-Programmen nicht immer der Fall ist. Sollten Sie nähere Informationen zu C wie auch zu den anderen Programmiersprachen benötigen, so sehen Sie sich am besten auf dem reichhaltigen Literaturmarkt um.

Die nächste Programmiersprache ist eine der umstrittensten. Es handelt sich dabei um Forth, das zu Beginn der siebziger Jahre von Charles H. Moore in den USA entwickelt wurde. Ursprünglich wurde es zur Steuerung

eines Radioteleskops entwickelt. Auf dieser Basis entstand eine sehr maschinennahe, schnelle und trotzdem noch relativ einfache Programmiersprache, die sich zu einem Standard auf dem Gebiet von Wissenschaft und Technik entwickelte.

Es gibt keine Programmiersprache, die so heiß diskutiert wird wie Forth. Durch die äußerst maschinennahe Programmierung erscheinen die Programme sehr unstrukturiert und unübersichtlich, doch die hohe Ablaufgeschwindigkeit kompensiert dieses Manko fast gänzlich.

Im Gegensatz zu vielen anderen Programmsystemen enthält Forth eine sehr interessante und praktische Programmablaufsteuerung. Ein Programm kann sowohl compiliert als auch durch den Interpreter ausgeführt werden. Die hohe Geschwindigkeit von Forth liegt zum großen Teil auch in seiner Arithmetikbehandlung begründet.

Die Arithmetikroutinen nutzen den Stapel genauso wie es die Rechenroutinen des Mikroprozessors machen. Der Wert, der als letzter auf dem Stapel abgelegt wurde, wird auch als erster wieder gelesen. Dieses Prinzip nennt man LIFO (Last in - First out).

Aus diesen den Maschinensprachenprogrammierern schon bekannten Datenstrukturen ergibt sich die umgekehrt polnische Notation (UPN) als Zahlenverarbeitung. Der Aufbau dieser Rechenweise ist schnell an einem Beispiel erklärt.

UPN und der Stapel

Normalerweise denken wir in einer Reihe von aufeinanderfolgenden Gedanken. Wollen wir zum Beispiel zwei Zahlen addieren, so übernehmen wir den Rechenterm nacheinander vom Anfang bis zum Ende in unser Gehirn. Dabei gehen wir beim Lesen der Formel sequentiell vor, das heißt, unser Auge wandert von der ersten Zahl bis zum Gleichheitszeichen. Tippen wir diese Rechenaufgabe in einen Taschenrechner, so übergeben wir wieder sequentiell die Rechung in den Taschenrechner. Wollen wir wissen, wieviel »11 + 12« ist, geben wir erst die Zahl 11 ein, dann das Plus-Zeichen, daraufhin die 12 und schließlich das Gleichheitszeichen. In der Anzeige erscheint dann die Zahl 23.

Im Gegensatz dazu legt Forth zuerst die Zahlen beziehungsweise Operanden, die es verknüpfen soll, auf dem Stapel ab. Trifft Forth dann auf den Verküpfungsfaktor - in unserem Fall das Plus-Zeichen - so holt er die letzten beiden Werte vom Stapel, verknüpft sie durch Addition und legt das Ergebnis schließlich wieder auf dem Stapel ab.

Forth wurde von einer Gruppe von Programmierern standardisiert und die »Forth Interest Group«, kurz FIG genannt, wacht über die Einhaltung. Trotzdem erfährt Forth immer noch Neuerungen und verbesserte Programmstrukturen, die es noch aktueller machen.

Die letzte Modifikation geht auf das Jahr 1983 zurück, in dem der letzte FIG-Standard festgelegt wurde. Hervorzuheben ist auch die erweiterbare Befehlsbibliothek von Forth. Das bedeutet, daß mit Hilfe der vordefinierten Prozeduren völlig neue Befehle beziehungsweise Funktionen definiert werden können. Hat man eine neue Funktion definiert, so nimmt Forth sie in sein sogenanntes Wörterbuch auf. Das Wörterbuch enthält alle Systemfunktionen und die selbstdefinierten Kommandos.

Durch diese Möglichkeit sind Sie in der Lage, den Sprachwortschatz und die Sprache selbst Ihren speziellen Wünschen anzupassen. Man definiert für die einzelnen Operationen entsprechende Worte, die genauso wie andere Forth-Wörter aufgerufen werden können.

Dabei spart man sich die Programmierung eines Hauptprogramms, das die Eingabekommandos interpretiert und die entsprechenden Unterprogramme aufruft. Das Forth-System erledigt dieses ganz automatisch und bietet gleichzeitig die Möglichkeit, in der neuen Kommandosprache zu programmieren. Das bedeutet, daß Sie sich unter Forth ein völlig neues Betriebssystem oder eine neue Programmiersprache schreiben können, die dann unabhängig benutzt werden kann.

Gespeichert wird ein Programm in sogenannten Screens, die aus je sechzehn Zeilen mit vierundsechzig Buchstaben bestehen. Diese Screens werden meistens gepackt gespeichert und dann bei Bedarf wieder geladen. Dabei erreicht Forth die große Geschwindigkeit durch die Compilierung des Quelltextes in Maschinensprache.

Ein großer Nachteil von Forth liegt in der groben Unstrukturiertheit, die ein übersichtliches Programmieren nicht gerade fördert. Wenn sich der Programmierer über längere Zeit ein Forth-Programm nicht mehr angesehen hat oder die Bearbeitung nach einem gewissen Zeitraum wieder fortsetzen will, so hat er kaum eine Chance, den ursprünglichen Quelltext wieder zu verstehen. Dieses Problem ergibt sich aus der speziellen Syntax der Forth-Programme und der ansonsten vorteilhaften Möglichkeit der Befehlsdefinition. Dieser Umstand erschwert natürlich auch die Fehlersuche.

Für die CPC-Reihe sind zwei Programme interessant, die seit einiger Zeit auf dem deutschen Markt angeboten werden. Es handelt sich dabei um das CPC-Forth von Holtkötter und den Public Domain-Compiler Forth 83. CPC-Forth wird entweder in der Kassettenversion oder in der Diskettenversion unter CP/M angeboten.

Forth gegen Forth

Der Befehlsumfang dieses Dialektes enthält zusätzlich zu den Grundbefehlen noch einige Kommandos, die die Arbeit mit dem System wesentlich erleichtern. Für Einsteiger ist CPC-Forth eine Alternative zu den anderen auf dem Markt befindlichen Forth-Programmen.

Allerdings kommt man um Sekundär-Literatur nicht herum. Ein umfangreiches Handbuch gehört zwar zum Lieferumfang, es wird aber seiner Aufgabe als begleitende Dokumentation nicht immer gerecht. So sollte zumindest ein kleines Kapitel für Einsteiger enthalten sein, da viele Anwender bestimmt ihre ersten Erfahrungen sammeln müssen. Doch in dem gesamten Begleitbuch ist nicht ein einziges Forth-Programm abgedruckt. Auch sind die Befehle der CPC-Forth-Version zu knapp erklärt; jeweils ein kurzes Beispielprogramm hätte Wunder gewirkt. Trotzdem ist Forth 139 Mark (auf Kassette nur 92 Mark) wert.

Im Gegensatz zum CPC-Forth läuft Forth-83 nur unter CP/M, das heißt, daß es keine Kassettenversion gibt. Doch die Leistungsdaten sind wirklich hervorragend. Forth-83 besitzt einen sehr umfangreichen Befehlssatz, der in keinerlei Hinsicht irgendwelche Schwächen hat. Hinzu kommen noch nützlich Hilfsprogramme wie ein Decompiler, ein Assembler und ein guter bildschirmorientierter Editor.

Forth-83 ist besser als viele andere kommerzielle Forth-Versionen und erleichtert dem Anwender den Einstieg in diese Programmiersprache sehr. Die Multitasking-Fähigkeit und viele CPC-spezifische Kommandos machen Forth-83 zu einem ohne Einschränkung zu empfehlenden Programm, das vor allem zu einem Preis von 30 Mark in jede Programmsammlung gehört.

Doch nun zu einer relativ exotischen Programmiersprache, die aber für

CPC-Besitzer ein breites und interessantes Aufgabengebiet erschließt. Es handelt sich um BCPL. BCPL ist eine Art Vorversion von C. BCPL wird oft als Sprache bezeichnet, die sich für die Systemprogrammierung eignet.

Diese Beschreibung wird dem Aufbau und der Funktionsweise dieser Programmiersprache jedoch nicht gerecht, denn BCPL kann mehr. Der besondere Vorteil von BCPL liegt in der Flexibilität. Programme, die in dieser Sprache geschrieben wurden, sind nicht durch Strukturierungsvorschriften beeinflußt, nach denen eine Variable nur einen bestimmten Typ repräsentieren darf.

Es gibt bei BCPL keine verschiedenen Variablentypen. Variablen werden einfach als Platzhalter für Zahlen verstanden und Operationen mit Zeichenketten werden zum Beispiel durch Manipulationen mit den numerischen Variablen erreicht. Diese Fähigkeit unterscheidet BCPL von vielen anderen Programmiersprachen, die eine exakte Variablendefinition benötigen. Eine Zeichenkette wird einfach durch eine Variable bestimmt, die auf den Anfang der Folge im Speicher zeigt. Selbst Prozeduren sind nur einfache Variablen, deren Wert die Lage einer Prozedur angibt.

Diese Flexibilität erlaubt es dem Programmierer, jeden gewünschten Bereich des Speichers (sogar jedes einzelne Bit) anzusprechen und somit mit Operationen zu arbeiten, die ansonsten nur Maschinensprache-programmen zugänglich sind.

Natürlich hat jeder Komfort seinen Preis. Das ist auch in BCPL nicht anders. Im Gegensatz zu anderen Programmiersprachen, die unsinnige oder fehlerhafte Routinen abfangen und anzeigen, muß BCPL hier passen. Der Programmierer ist in BCPL sehr frei und darf auch in sein eigenes Verderben laufen.

Ein BCPL-Compiler für die CPCs ist von dem britischen Softwarehaus Arnor als EPROM-Version mit Diskette für 129,90 Mark erhältlich. Auf der Diskette befinden sich beispielsweise zwei verschiedene Demonstrationsprogramme, die zwei völlig unterschiedliche Anwendungsbereiche ansprechen. Beide überraschen durch ihre Geschwindigkeit. Da gibt es einerseits einen Disassembler und andererseits – man höre und staune – ein akzeptables Space Invaders. BCPL für die Schneider CPC-Computer enthält den vollen BCPL-Standardbefehlssatz und ist als interessantes exotisches Programm sehr zu empfehlen.

Mit der nächsten Programmiersprache nähern wir uns dem absoluten Optimum an Geschwindigkeit. Es geht um Assembler beziehungsweise Maschinensprache. An der Maschinensprache führt kein Weg vorbei, denn jeder Compiler übersetzt ein Programm letzten Endes in Maschinensprache.

Die Assembler-Routinen geben dem Mikroprozessor und den zugehörigen Bausteinen direkt Kommandos, die innerhalb weniger Mikrosekunden ausgeführt werden. Die Maschinenspracheroutinen werden sofort ausgeführt.

Der eigentliche Quelltext muß vorher durch einen Editor geschrieben und mit Hilfe eines Assemblers übersetzt werden. Der Assembler überträgt dann jeden einzelnen Befehl in eine für den Mikroprozessor (auf den CPCs der Z80) verständliche Byte-Folge von höchtens vier Byte.

Für die CPC-Computer von Schneider existieren zwei interessante Assembler-Programme, die einen unterschiedlichen Anwenderkreis ansprechen.

Das erste System ist das Assemblerpaket aus der Public Domain-Software zu 30 Mark. Es enthält einen Assembler, einen intelligenten Disassembler sowie einen Maschinensprachemonitor, der leicht zu handhaben ist. Obwohl das gesamte Paket unter CP/M läuft, benutzt der Assembler den Register- und Befehlssatz des Z80-Mikroprozessors. Neben dem leistungsfähigen Assembler befinden sich auf der Diskette noch ein Linker, der die vorhandenen Programm-Module zusammenfügt, und ein Monitor, der sich auch zur Einzelschrittbearbeitung eines Maschinensprache-programms eignet.

Schlaues Paket

Der intelligente Disassembler besticht auch durch umfangreiche Manipulationsmöglichkeiten. Der mitgelieferte Assembler eignet sich gut zu einem modularen Aufbau der einzelnen Programmteile. Er erzeugt aus jeweils fehlerfreien Quelltexten einen Code, der sich gut in andere Programmteile einbinden läßt.

Diese Modultechnik hat zur Folge, daß bei der Assemblierung eines Quelltextes nicht alle Routinen übersetzt werden müssen und dadurch viel Zeit und Fehlerbehandlung erspart bleibt.

Der Disassembler hebt sich deutlich von den allgemein bekannten Disassemblern ab, die Maschinencode einfach in Mnemonics zurückwandeln. Der Disassembler dieses Pakets erkennt sogar automatisch Wortketten von mehr als acht Zeichen Länge und markiert alle in einem Programm vorkommenden Sprungmarken.

Der Monitor ist schließlich der letzte Bestandteil diese Pakets. Er enthält einen kompletten Z80-Assembler und gleichzeitig eine Breakpoint-Verwaltung von bis zu sechzehn Breakpoints.

Als ROM-Version für die CPCs unter Amsdos empfiehlt sich das Maxam-Modul, das zusätzlich zu den normalen Funktionen einen äußerst komfortablen Editor enthält. Es ist einer der schnellsten Assembler, die es für den Schneider-CPC gibt. Die EPROM-Version kostet 129,90 Mark und die Diskettenversion 99,90 Mark.

Nach den geschwindigkeitsorientierten Sprachen wenden wir uns nun den Dialekten zu, die für ein besonderes Anwendungsgebiet geschaffen wurden. Die erste der wichtigen Programmierumgebungen ist Cobol (Common business oriented language). Cobol ist im Bereich der kaufmännischen und Wirtschaftsprogramme die Programmiersprache, die man am häufigsten antrifft. Bereits im Jahr 1960 erschien die erste Version.

Da die damals verwendeten Programmiersprachen wie Algol oder Fortran die in sie gesetzten Erwartungen nicht mehr erfüllen konnten, wurde Cobol entwickelt. Hervorzuheben sind die Möglichkeiten zur Dokumentation und die gute Lesbarkeit des Programmquelltextes. Ein Cobol-Programm liest sich wie englische Prosa, die beschreibt, was getan wird. Aber in der guten Dokumentation von Cobol liegt auch ein ziemlich großer Nachteil. Da ein Programm aus vier Hauptteilen besteht (Identification Division, Environment Division, Data Division und Procedure Division), muß der Einführungs- und Variablendeklarationsteil jeweils wiederholt werden.

Die Nevada-Cobol-Version für den CPC entspricht fast völlig dem ANSI-Standard von 1974 und enthält für 99 Mark auch sinnvolle Befehlserweiterungen, die die Programmierung komfortabler gestalten.

Der Vater aller Programmiersprachen ist Fortran (formula translator). Fortran wurde hauptsächlich für Anwendungen im wissenschaftlichen und technischen Bereich konzipiert und findet dort auch heute noch Einsatz.

Seit dieser Dialekt erstmals im Jahre 1966 ANSI-standardisiert wurde, entpuppte er sich schnell als beliebte Programmierumgebung. Konzeptionell herrscht in Fortran das Prinzip des modularigen Aufbaus vor, so daß der heutige Fortran-Program-

mierer auf eine große Anzahl von Standardroutinen und -utilities zurückgreifen kann.

Die vorliegende, nur unter CP/M lauffähige Version namens Nevada-Fortran enthält zwar nur eine Untermenge der im ANSI-Standard festgelegten Kommandos, jedoch eignet sie sich trotzdem für kleine bis mittlere Anwendungsbereiche. Kostenpunkt: 99 Mark.

Letztendlich zu den Sprachen der sogenannten »Künstlichen Intelligenz«, kurz KI. Für die Schneider-CPCs gibt es jeweils zwei interessante Versionen, die sich für den Einstieg in dieses Gebiet eignen.

Ein professionelles System für den Schneider CPC ist micro-Prolog. Das gelieferte Programmpaket enthält neben der Programmdiskette ein umfangreiches Prolog-Handbuch, das alle Besonderheiten von micro-Prolog beschreibt. Lobenswerterweise liegt dem Programm aber zusätzlich – übrigens als einzigem Programm in unserem Test – noch ein Handbuch bei, das alle Grundlagen von Prolog erklärt und damit den Einstieg in die Programmierung erleichtert.

Lisp ist ebenfalls wie Forth auf der umgekehrt polnischen Notation aufgebaut. Da Lisp sehr flexibel ist, wird es auch heute noch benutzt, obwohl seine Grundzüge schon 1959 festgelegt wurden. Es ist vorwiegend zur Listenverarbeitung geschrieben worden und ähnelt damit in gewisser Weise der wesentlich aktuelleren Sprache C.

Als Public Domain-Software sind sogar beide Sprachen zusammen auf einer Diskette gespeichert. Es ist aber anzumerken, daß diese Versionen der beiden Sprachen nur den Grundwortschatz des jeweiligen Programmiersystems implizieren und aus diesem Grund fast ausschließlich für kleinere Programme geeignet sind.

Da das Public Domain-Programm XLisp (30 Mark) aber den Grundwortschatz mit Möglichkeiten zur objektorientierten Programmierung verknüpft, benötigt diese Version mindestens 128 KByte Speicherplatz. Der Prolog-Interpreter E-Prolog für ebenfalls 30 Mark gibt sich hingegen wegen seiner Länge von nur 6 KByte auch mit weniger Speicherplatz zufrieden.

Jedes der vorgestellten Programme hat sein bestimmtes Einsatzgebiet. Darüber sollten Sie sich zuerst im klaren sein, bevor Sie ein Programm kaufen. Generell bieten die aufgeführten Public Domain-Programme eine hervorragende Alternative zu den kompletten und umfangreichen professionellen Programmiersystemen. Diese Programmsprache-Versionen sind vor allem dann wichtig, wenn man sich noch nicht entschieden hat, welche Sprache sich für die eigenen Zwecke eignet.

In vielerlei Hinsicht ist die Public Domain-Software sogar besser als die professionell angebotene. Manche Dialekte, wie zum Beispiel Forth-83, stellen sogar die optimale Programmierumgebung für die spezielle Sprache dar.

Sollte die Public Domain-Version allerdings einen in Anbetracht des festgelegten Sprachenstandards wesentlich geringeren Befehlsumfang besitzen, so ist bei weiterführendem Interesse die professionellere Version zu empfehlen, die normalerweise auch ein umfangreicheres Utility-Paket enthält. (Markus Zietlow/gn/ma)

Bezugsquellen:
Assembler-Paket, Xlisp, E-Prolog, Small-C, Forth-83: Martin Kotulla, Grabbestr. 9, 8500 Nürnberg 90
Turbo-Modula: Heimsoeth Software, Fraunhoferstr. 13, 8000 München 5
Small-C: Markt&Technik Verlag, Hans-Pinsel-Str. 2, 8013 Haar
CPC-Forth: Holtkötter, Albert-Schweizer-Ring 9, 2000 Hamburg 70
BCPL, Maxam: Arnor Deutschland, Hans-Henry-Jahn-Weg 21, 2000 Hamburg 76
Nevada-Cobol, Nevada-Fortran: Comfood, Rohrbusch 79, 4400 Münster-Roxel
Micro-Prolog: Brainware, Gustav-Mayer-Allee 25, 1000 Berlin 65

Einer kam durch

Professionelle Kopierprogramme versprechen viel. Doch was vermögen sie wirklich zu leisten? Wir ließen für Sie die drei interessantesten Kandidaten gegeneinander antreten. Eins ist fast perfekt.

Nahezu jeder Besitzer eines Schneider-Computers ist schon einmal in die Situation geraten, von einem Programm eine Kopie anfertigen zu müssen. Ist das Programm als Datei auf Diskette gespeichert, benutzt man in der Regel die mit dem Laufwerk gelieferte Utility »Filecopy«. Für ganze Disketteninhalte bietet sich dagegen das ebenfalls mitgelieferte »Disccopy« oder »Diskit« an. Beide Hilfsprogramme eignen sich für sämtliche ungeschützte Software. Was aber tun, wenn das Original kopiergeschützt ist? Da helfen die Programme aus dem Lieferumfang absolut nicht weiter. Doch warum sollte man überhaupt Sicherheitskopien anlegen, zumal doch schon der Text auf der Verpackung des teuren Programms vor unerlaubtem Kopieren und dessen rechtlichen Konsequenzen ausdrücklich warnt?

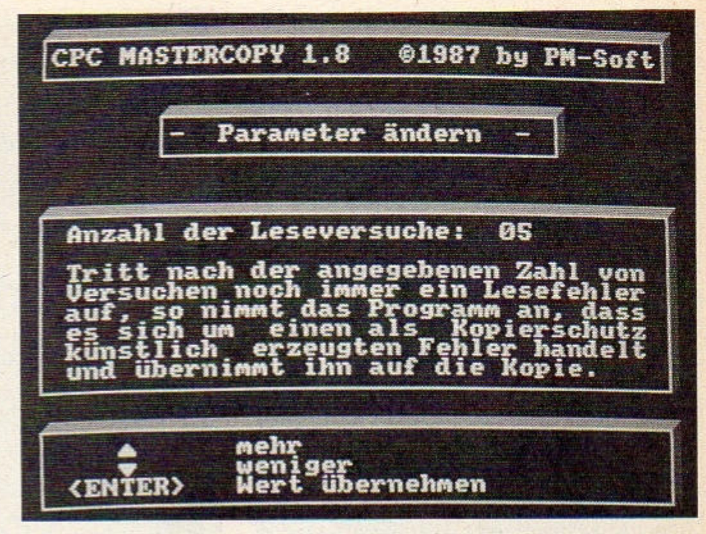
Die Frage beantwortet sich wie von selbst, wenn die Diskette mit der betreffenden Software eines schönen Tages verdächtig im Laufwerk zu rattern beginnt, weil erste Lesefehler auftreten. Hat die Diskette erst einmal Schaden genommen, kommt man meist nicht umhin, das Programm erneut zu kaufen. Einige wenige Software-Hersteller bieten für solche Fälle zwar eine Art Umtausch-Service an. Umstände wie längere Wartezeiten, während derer man das Programm ja nicht nutzen kann, und unverhältnismäßig hohe Kosten für die Kopie (inklusive Porto und Kostenbeteiligung kommt man leicht auf 30 Mark oder mehr, während selbst teuerste Leerdisketten im Laden nebenan nicht mehr als 10 bis 12 Mark kosten) lassen diesen »Service« jedoch nur als schwachen Trost erscheinen.

Andere Softwarefirmen profitieren von diesem zwangsläufigen Interesse der Computerbenutzer an Sicherheitskopien, indem sie spezielle Kopierprogramme anbieten. So wird der Markt in letzter Zeit von unzähligen derartigen Utilities geradezu überschwemmt. So gibt es Kopierprogramme, die geschützte Kassettensoftware auf Diskette kopieren (oder zumindest vorgeben, es zu vermögen). Dazu kommen die recht teuren Kopiermodule »Disc-Wizard«, »Multiface Two« und »Mirage Imager«, bei denen man nur noch auf einen Knopf drücken muß, um das Programm im Arbeitsspeicher auf Diskette zu speichern (mehr dazu erfahren Sie aus unserem Test auf Seite 11).

Den größten Marktanteil bestreiten Diskettenkopierprogramme. Als normaler Computerbesitzer steht man etwas hilflos vor diesem mannigfaltigen Angebot, da alle diese Produkte überragende Fähigkeiten versprechen, es aber »schwarze Schafe« darunter gibt, die nur einen verschwindend geringen Teil geschützter Programme kopieren.

Um zu verstehen, warum einige Diskettenkopierprogramme mehr leisten als andere, müssen wir uns zunächst

Supercopy informiert während des Kopierens stets über den Stand der Dinge



Die Parameter von Mastercopy lassen sich jedem Kopier-Problem anpassen

die Frage stellen, wie eine Diskette überhaupt zu schützen ist. Umfassendes Wissen darüber erwerben Sie durch die Lektüre des Beitrags »Schutz – wie lange noch?« auf Seite 45. Wollen Sie nicht so weit in diese Materie einsteigen, bieten wir Ihnen hier einen kleinen Abriß der wichtigsten Details zum besseren Verständnis des Tests.

Normalerweise besteht jede formatierte CPC-Diskette aus 40 Spuren. Einige Softwarefirmen formatieren die Disketten durch einen Trick mit bis zu 43 (3-Zoll-Laufwerke der ersten Generation verarbeiten sogar bis zu 44 Spuren) Spuren und verstecken ihren Kopierschutz auf einer dieser zusätzlichen Spuren. Spuren können Sie sich als eine Art magnetische »Fahrbahn« vorstellen, an denen sich der Lesekopf des Diskettenlaufwerks orientiert. Jede Spur wiederum ist in eine Anzahl Sektoren unterteilt. Diese Sektoren tragen die eigentlichen Informationen, die der Computer in seinen Arbeitsspeicher lädt. Die Zahl der Sektoren einer Spur beträgt normalerweise 9, manchmal ist sie aber bei kopiergeschützten Disketten manipuliert und dementsprechend höher oder niedriger. Jeden Sektor erkennt und unterscheidet der Lesekopf an der sogenannten Sektor-ID (ID=Identifikation). Die Sektor-ID enthält Informationen über Sektorlänge, Lesekopfnummer, Spurnummer und Sektornummer. Sie dient dem Computer auch zur Erkennung, ob der Sektor formatiert ist, also überhaupt Daten trägt.

Wie lassen sich also Disketten gegen das Kopieren schützen? Zum Leidwesen der Kopierwilligen gibt es da eine – wenn auch endliche – Vielzahl von Möglichkeiten:

- Manipulation der Sektoranzahl pro Spur
- Veränderung der Sektorgrößen
- Sektoren mit gelöschter Data-Adress-Mark
- Sektoren mit illegaler Sektor-ID (zum Beispiel falsche Spur- beziehungsweise Kopfnummer oder unrealistische Sektorgröße)
- künstliche Lesefehler (beispielsweise Prüfsummenfehler)
- unformatierte Spuren
- gleiche Sektor-IDs
- physikalische Verschiebung des ersten Sektors

Vielschichtige Kopier-Hindernisse

Diese Liste ist keineswegs vollständig, da die Softwarefirmen ständig an neuen Schutzmechanismen tüfteln. Sie gibt uns jedoch zuverlässige Kriterien in die Hand, um die Fähigkeiten unserer Probanden auszuloten und zu beurteilen.

Das Ziel eines Kopierprogramms ist, möglichst viele Kopierschutzmechanismen zu »überlisten«, indem es sie einfach mitkopiert. So leicht, wie sich das im ersten Moment anhören mag, gestaltet sich dieses Unterfangen indes keineswegs. So gelingt es diversen Kopierprogrammen beileibe nicht, eine zufriedenstellende Zahl von Schutzmechanismen außer Kraft zu setzen. Sie können sicher den Ärger eines Anwenders nachfühlen, der ein nicht gerade billiges Diskettenkopierprogramm erwirbt und dann feststellen muß, daß sich damit nur der geringste Teil seiner Programmsammlung kopieren läßt.

Daher wollen wir für Sie die Spreu vom Weizen trennen, indem wir Ihnen drei aktuelle und – glaubt man der Werbung – leistungsfähige Kopierprogramme vorstellen und sie in einem Vergleichstest antreten lassen.

Die Bezeichnung Clone (deutsch: Klon) bedeutet soviel wie »identische Kopie« und ist zur Zeit im gentechnischen Bereich in aller Munde. Ein Kopierprogramm diesen Namens erschien erstmals Ende 1986 auf dem Software-Markt. Es stammt vom Entwickler Ralf Waldeck, der schon seit zwei Jahren auf dem CPC 664 programmiert, nachdem er seines ZX81 überdrüssig war. Der Entwickler erklärt die Funktionsweise seines Programms, freilich ohne seine Geheimnisse preiszugeben, wie folgt: »Zuerst liest Clone eine Sektor-ID, um an der Sektornummer zu erkennen, ob die Diskette formatiert ist. Ist das nicht der Fall, bricht Clone den Kopiervorgang sofort ab (wer kopiert schon unformatierte Disketten?). Alle Sektoren werden so auf ihre Formatierung geprüft. Jeweils acht formatierte Spuren werden in den Arbeitsspeicher eingelesen. Der Benutzer legt nun eine Diskette ein, auf die das Programm kopiert werden soll und die acht Spuren werden geschrieben. Dieser Vorgang wiederholt sich einige Male, bis die ganze Diskette kopiert ist.«

Ein Schnellader sorgt für die kurze Ladezeit von Clone, nach deren Ende man vor der Wahl steht, mit einem oder zwei Laufwerken (mit zwei Laufwerken ist die Kopierrichtung immer: von A nach B) 40 oder 44 Spuren zu kopieren. Beim erfolgreichen Kopieren entstehen exakte 1:1-Kopien; die Kopie unterscheidet sich also in nichts vom Original.

Auch Supercopy arbeitet mit ein oder zwei Laufwerken, bietet dafür jedoch im Gegensatz zu Clone vier wahlfreie Kopierrichtungen (von Laufwerk A nach Laufwerk A, von B nach B, von A nach B und von B nach A). Es arbeitet unter Amsdos und kümmert sich um 43 Spuren. Da die neueren 3-Zoll-Laufwerke nur 43 Spuren verar-

Kopierschutz	Clone	Supercopy	Mastercopy
Manipulation der Sektorzahl	X	X	X
Veränderung der Sektorgröße	X	X	X
gelöschte Data-Adress-Mark		X	X
Sektoren mit fehlerhafter ID	X	X	X
künstliche Lesefehler			X
unformatierte Spuren	X	X	X werden auf Zieldiskette »entformatiert«
physikalische Umpositionierung von Sektoren			X prüft die physikalische Lage der Sektoren und, ob der erste Sektor auch in Höhe des Indexlochs steht
leiche IDs	X sehr unsicher	X unsicher	X sicher

Diese Schutzmechanismen verarbeiten die drei Prüflinge

beiten (manche gar nur 42), genügt diese Spanne in der Praxis völlig.

Zwei Maschinenbau-Studenten, R. Günther und Th. Scholl, begannen im Mai des Jahres 1986 mit der Entwicklung von Supercopy: »Wir wollten uns einmal mit der Programmierung in Maschinensprache bekanntmachen. Als erstes Projekt bot sich uns ein Kopierprogramm an, da es seinerzeit nichts Vergleichbares gab«. Nach intensiver Lektüre einschlägiger Literatur und einem Vierteljahr angestrengter Arbeit war dann Supercopy fertig. Tests mit insgesamt 500 Programmen fielen zur vollsten Zufriedenheit der Entwickler aus.

Da man aber nie weiß, was sich die Software-Schützer demnächst für neue Tricks einfallen lassen, gibt es für Supercopy einen sogenannten Update-Service. Dieser beinhaltet den kostenlosen Umtausch des gekauften Supercopy innerhalb kürzester Zeit in eine neue, erweiterte Version mit der Fähigkeit, den neuen Kopierschutz zu verarbeiten. Dazu sendet der Käufer die Diskette mit dem »Problemfall« und sein Supercopy-Original an den Hersteller.

Mastercopy wurde von dem 17jährigen Schüler Peter Mandrella programmiert. Es entstand aus einer Art Notsituation heraus: »Ich wollte Sicherheitskopien von einigen meiner Spielprogramme anfertigen, mußte aber leider feststellen, daß alle verfügbaren Kopierprogramme jämmerlich versagten. Also kam mir die Idee, selbst ein solches Programm zu schreiben und ich machte mich ans Werk.«

Was dabei herauskam, kann sich wirklich sehen lassen. Für das Kopieren einer ganzen Diskettenseite braucht man im günstigsten Fall mit einem Laufwerk etwa 1½ Minuten, zwei Laufwerke erledigen die Aufgabe gar in nur knapp einer Minute. Beim Bearbeiten der Zieldiskette wird im selben Schritt formatiert und geschrieben. Viele andere Programme erledigen das in zwei getrennten Schritten.

Mastercopy verfügt als einziges der getesteten Utilities über einen internen Timer, der bei gleichen Sektor-IDs für den Zugriff auf den richtigen Sektor sorgt. Auch mit Mastercopy lassen sich komfortablerweise bei Verwendung zweier Laufwerke Quell- und Ziellaufwerk nach Belieben auswählen. Durch eine wahlweise Geschwindigkeitsoptimierung ist die Kopiergeschwindigkeit zu beeinflussen. Das erweist sich in der Praxis als überaus nützlich, da viele Programme mit dieser höheren Geschwindigkeit zu kopieren sind. Alternativ steht eine Sicherheitsoptimierung zur Wahl, mit der sich die maximale Kopiersicherheit variieren läßt. Mastercopy testet dann nicht auf alle Schutzmechanismen. Eine Erhöhung geht selbstverständlich zu Lasten der Geschwindigkeit. Auch andere Parameter wie zum Beispiel die Anzahl der Leseversuche sowie erste und letzte Spur sind variabel einzustellen.

Der Hersteller versichert auf der Packung, daß »Mastercopy alle zur Zeit auf dem Markt befindlichen Programme (Stand März 1987) kopiert«. Und treten irgendwann einmal Schwierigkeiten beim Kopieren eines neuen Programms auf, existiert für Mastercopy ein ähnlicher Update-Service wie für Supercopy. Allerdings kostet jeder Umtausch in eine neue Version jeweils 10 Mark.

Für den Vergleichstest sind die drei Kriterien Kopiersicherheit, Benutzerfreundlichkeit und Preis ausschlaggebend. An diese Reihenfolge der Gewichtung wollen wir uns halten.

Um die Kopiersicherheit festzustellen, haben wir keine Mühen gescheut und versucht, 100 Programme, ältere und neuere Spiel- beziehungsweise Anwendersoftware, mit jedem der drei Probanden zu kopieren. Das Ergebnis rechtfertigte unsere Anstrengungen, denn es offenbarte Erstaunliches.

Clone kopierte zunächst »wie eine Eins« und wir fragten uns, ob wir mit der Zahl der Testdurchgänge nicht doch etwas zu hochgegriffen hatten. Doch – kaum waren diese ketzerischen Gedanken zu Ende gedacht – zeigten sich die ersten »Aussetzer«. Nach dem hundertsten Haken auf der ersten Checkliste stand dann das Endergebnis fest: Clone verarbeitete insgesamt 79 Prozent der Programme erfolgreich. Zur Ehrenrettung müssen wir Clone aber zugestehen, daß es zu den etwas gesetzteren Semestern zählt und seine »Fehlzündungen« sich auf relativ neue Programme beschränken. Dieser Umstand wäre nicht so tragisch, böte die Firma Waldeck einen Update-Service wie ihre Konkurrenten.

Dreikampf

Durch das etwas magere erste Teilergebnis etwas entmutigt, wandten wir uns dem zweiten Durchgang zu. Um es kurz zu machen: Supercopy schlug sich wacker und kopierte 92 Prozent der Software. Es scheiterte an Programmen diverser Labels, was nur zum Teil daran lag, daß es einzelne

In letzter Minute

Wie eine telefonische Nach-Recherche bei den Herstellern der Programme Supercopy und Mastercopy ergab, sind mittlerweile die im Test erwähnten fehlenden Schutzmechanismen von beiden Utilities kopierbar. Supercopy soll nun auch Mastercopy problemlos kopieren und selbst einen verbesserten Kopierschutz erhalten haben, den Mastercopy wiederum nicht verarbeitet. Am Testergebnis ändert dieser Umstand indes nichts, da man stets nur vom Status Quo ausgehen kann. Da aber bekanntlich der Update-Service existiert, ändert sich die Situation laufend. Nachdem nun also beide angeblich sämtliche derzeit gebräuchlichen Schutzmechanismen kopieren, liefern sich ihre Programmierer einen einsamen Wettkampf...

Kopierschutzmechanismen nicht überlisten konnte. Vielmehr kopierte es manchmal erstaunlicherweise auch solche Programme nicht, für deren Kopierschutz es eigentlich gerüstet ist; es fabrizierte schlicht Fehler beim Schreiben der Daten. So greift Supercopy bei Sektoren mit gleicher ID willkürlich auf einen der Sektoren zu. Dadurch geraten die Sektoren mitunter in eine falsche Reihenfolge. Auf diese Weise entspricht der Aufbau der Zieldiskette nicht exakt dem des Originals und es kommt zu empfindlichen Fehlern.

Ganz anders liegt der Fall bei Mastercopy, an dem sich unsere Tester die Zähne ausbissen. Der Hinweis auf der Verpackung, »Mastercopy kopiert alle zur Zeit auf dem Markt befindlichen Programme«, animierte uns natürlich, das Programm besonders kritisch unter die Lupe zu nehmen und diese Aussage gegebenenfalls zu widerlegen. Nachdem bereits alle 100 Testprogramme erfolgreich ihren Weg auf vormals leere Disketten gefunden hatten, packte uns der Ehrgeiz: Es mußte doch irgend etwas zu finden sein, das Mastercopy in seine Schranken verweist. Also schafften wir weitere Programme heran. »110«, »120«, »130« und zuletzt »134« vermerkten die Tester als Zahl der Testdurchläufe – keine Chance! Am Testergebnis änderte sich nicht das geringste, denn mehr als 100 Prozent sind naturgemäß nicht möglich. Mastercopy triumphierte also souverän über all unsere Bestrebungen. Es kopiert wirklich alles, was man ins Diskettenlaufwerk schiebt – solange es zumindest annähernd einer Diskette ähnelt. Aber Spaß beiseite. Bei allem Überschwang der Gefühle muß man gestehen, daß es derzeit doch zumindest ein Programm gibt, das mit der jetzigen Mastercopy-Version nicht kopierbar ist. So jedenfalls lautet die erfreulich ehrliche Auskunft der Firma Computer-Corner – Nobody is perfect. Doch der Programmierer Peter Mandrella setzt derzeit alle Anstrengungen daran, auch den Kopierschutz für eine neue Mastercopy-Version zu analysieren. Diese neue Version ist dann von jedem registrierten Käufer beim Hersteller im Umtausch zu beziehen.

Als krönenden Abschluß des Tests ließen wir die drei Konkurrenten nochmals gegeneinander antreten; diesmal jedoch im direkten Vergleich. Das bedeutet, wir haben mit jedem der Kopierprogramme versucht, die jeweils anderen zu kopieren. Das Ergebnis unterstreicht die bereits gesammelten Erkenntnisse: Mastercopy kopiert sowohl Clone als auch Supercopy, Supercopy kopiert nur Clone, Clone aber versagt seine Dienste in beiden Fällen.

Längst nicht so viel Bedeutung wie der Kopiersicherheit kommt dem Kriterium Benutzerfreundlichkeit bei unserer Bewertung zu. Clone schneidet auch hier am schlechtesten ab. So fragt Clone, wie oben bereits erwähnt, zum Beispiel am Anfang, ob es 40 oder 44 Spuren kopieren soll. Woher soll aber der Benutzer wissen, ob die Originaldiskette nun gerade 40 oder 44 Spuren trägt? Diese Arbeit sollte ja wohl eigentlich Aufgabe des Kopierprogramms sein. Als zweiten und gravierendsten negativen Punkt werteten wir Clones Unvermögen, die Kopierrichtung bei der Verwendung von zwei Laufwerken frei zu wählen. Gerade wenn man ein Zweitlaufwerk mit anderem Diskettenformat (3½ oder 5¼ Zoll) betreibt, ist eine solche Auswahl wichtig, um Konvertierungen zwischen diesen verschiedenen Formaten zu erlauben.

Supercopy stellt von der Bedienerfreundlichkeit quasi das Nonplusultra dar, denn die Eingaben des Benutzers beschränken sich auf das äußerste erreichbare Minimum. Man muß es wirklich nur noch starten und die Kopierrichtung wählen.

Ähnlich verhält es sich mit Mastercopy. Es ist von vornherein auf 42 Spuren eingestellt. Falls einmal ein Kopierversuch nicht die gewünschten Erfolge zeigt, kann der Benutzer einen erneuten Versuch mit einer anderen Spuranzahl starten. Es erlaubt die freie Wahl der Kopierrichtung und verfügt außerdem über eine wahlweise Geschwindigkeits- oder Sicherheitsoptimierung. Diese zusätzliche Ausstattung bedingt natürlich auch zusätzliche Handgriffe.

Die Anleitung ist bei Mastercopy und Clone ins Programm integriert, während sie der Supercopy-Käufer auf einem Beipackzettel findet.

Doppelbödige Moral

Clone und Mastercopy lassen sich nicht mit sich selbst kopieren, so daß der Benutzer auf Sicherheitskopien dieser Programme verzichten muß. Es bleibt jedem Leser überlassen, sich seine eigenen Gedanken ob solcher Doppelmoral zu machen. Als einziges Programm im Testfeld gesteht Supercopy seinem Besitzer genau eine Sicherheitskopie von sich selbst zu. Beim Versuch, eine zweite Kopie anzulegen, macht das Programm freundlich darauf aufmerksam, daß eine Kopie ja wohl genüge. Dieser Weg ist sicher eine sehr gute Lösung.

Clone	Plazierung
Kopiersicherheit	3 (79%)
Benutzerfreundlichkeit	2
Preis	1 (zirka 60 Mark)
Gesamtwertung	3
Supercopy	**Plazierung**
Kopiersicherheit	2 (92%)
Benutzerfreundlichkeit	1
Preis	3 (zirka 80 Mark)
Gesamtwertung	2
Mastercopy	**Plazierung**
Kopiersicherheit	1 (100%)
Benutzerfreundlichkeit	1
Preis	2 (zirka 70 Mark)
Gesamtwertung	1

Einerseits hat nämlich der Benutzer eine kostenlose Sicherheitskopie von Supercopy, andererseits lassen sich so keine Raubkopien anfertigen. Eine Sicherheitskopie von Clone ist gegen eine Gebühr von 20 Mark beim Hersteller erhältlich. Das schlägt sich natürlich negativ auf den effektiven Endpreis nieder.

Clone ist mit seinem Preis von etwa 60 Mark zwar in der Anschaffung billiger als die beiden anderen Programme, die zirka 70 (Mastercopy) beziehungsweise 80 Mark (Supercopy) kosten, aber angesichts der Kopierleistungen sollte man ruhig mehr auf den Ladentisch legen, um ein zuverlässigeres Programm zu erwerben. Vom idealen Kopierprogramm sind wir leider immer noch ein Stück entfernt. Es müßte eine Kombination der positiven Eigenschaften bieten, also absolut alles kopieren und dabei, ohne Dazutun des Benutzers, vollautomatisch vorgehen. Wollen wir gemeinsam hoffen, daß wir Ihnen bald ein solches Programm an dieser Stelle vorstellen können. Vielleicht entwickelt sich ja eins der beiden Top-Programme durch die ständigen Verbesserungen dorthin.

Zu einem Test mit derartigem Inhalt gehört als Abschluß selbstverständlich der Hinweis, daß gemäß § 53/IV des Urheberschutzgesetzes (UrhG) »die Vervielfältigung eines Programms für die Datenverarbeitung oder wesentliche Teile davon nur mit Einwilligung des Berechtigten zulässig« ist. Aber eigentlich nutzen Sie ja ohnehin keins der Programme, um Raubkopien anzufertigen – oder?

(Carsten Bormeier/ja)

Clone: Waldeck Software, Tulpenstraße 30, 2870 Delmenhorst
Mastercopy: Computer Corner, Taubentränke 14, 5470 Andernach
Supercopy: Weeske Software, Wiflingshauser Straße 83, 7300 Esslingen am Neckar

Para 3.0, Diskpara und

Hatten Sie nicht auch schon mal das Problem, Daten zwischen dem Schneider CPC und einem anderen Computersystem auszutauschen? Die Lösung sind Programme wie Para 3.0 und Diskpara. In diesem Vergleichstest zeigen wir Ihnen, was die beiden leisten.

Wer sich intensiv mit CP/M beschäftigt, weiß, daß es eine Vielzahl unterschiedlicher Aufzeichnungsformate für Disketten gibt. Der Datenaustausch wird dadurch natürlich erschwert und man muß sich schon einige Tricks einfallen lassen, um überhaupt zu einem Erfolg zu kommen. Im CP/M befinden sich zum Beispiel Übersetzungstabellen wie der Diskparameter-Block (DPB), in dem alle Details (Parameter) des gewünschten Formats eingetragen sind. Will man nun das Aufzeichnungsformat ändern, dann braucht man ein Programm, das die Eintragungen im DPB ändert. Programme, die genau dies erlauben, sind Para 3.0 und Diskpara.

Von diesen beiden Programmen ist Para 3.0 neu auf dem Markt. Es ist schon eine Weile her, daß die erste Version dieses Vortex-Programms erschien. Jene erste Version wies Kinderkrankheiten auf, die offenbar schwer behebbar waren. Einer der am häufigsten bemängelten Fehler war, daß Para die Zusammenarbeit mit dem VDOS des X-Moduls verweigerte. Doch diese Krankheiten sind nun behoben. Und bei dieser Gelegenheit erfuhr das Programm auch gleich eine erhebliche Erweiterung seiner Leistungsfähigkeit.

Im Lieferumfang enthalten ist ein sehr umfangreiches Anleitungsbuch, das die Menüführung anhand von Bildschirmausdrucken sehr anschaulich erklärt. Der Umfang der Funktionen von Para 3.0 erfordert eine Aufteilung der Benutzerführung auf mehrere Bildschirme. Das geht allerdings auf Kosten der Übersichtlichkeit. Nach einer Eingewöhnungszeit ist das aber rasch vergessen. Eine wichtige Funktion ist die, bis zu vier virtuelle (logische) Laufwerke zu simulieren sowie deren Parameter bis zum nächsten Kaltstart ins BIOS resistent einzubinden. Im Klartext heißt das: Sogar nach Verlassen von Para 3.0 stehen mehrere Formate zur Verfügung, die man wie auf echten Laufwerken anspricht. Die wählbaren zusätzlichen Laufwerke und die Kurzbezeichnung der dort erreichbaren Formate ist bei jedem Warmstart als BIOS-Meldung am oberen Bildschirmrand eingeblendet. Getrübt wird die Freude allerdings durch einen um 10 KByte verkleinerten Arbeitsspeicher (TPA), was beim Betrieb ohne Vortex-Speichererweiterung den Ablauf einiger Programme nicht mehr zuläßt. Dazu gehören vor allem Turbo-Pascal-Programme, da der Compiler sehr verschwenderisch mit dem Speicherplatz der TPA umgeht, sowie einige Utilities, die das BIOS aufrufen und sich aufgrund der verschobenen Adressen nicht mehr zurechtfinden. Para 3.0 unterstützt bis jetzt nur CP/M 2.2. Eine runde Sache wird Para 3.0 erst in der fortgeschrittenen Anwendung, am besten mit der Vortex-Doppelstation. Dann kann man nämlich in Laufwerk A die Para 3.0-Diskette und in Laufwerk B die Fremd-Diskette legen.

Das von Para 3.0 aus zu startende Analyse-Programm erlaubt eine umfassende Untersuchung der Disk-Parameter, sowie der Spur- und Sektorübersetzung, die tabellarisch übersichtlich dargestellt werden. Speichert man einige Einstellungen, so wird beim Neustart von Para 3.0 automatisch die Voreinstellung sichtbar und sie läßt sich direkt mit dem RAM-BIOS-Aufruf einbinden. Dann kehrt Para 3.0 sofort zu CP/M zurück, die simulierten logischen Laufwerke bleiben erhalten.

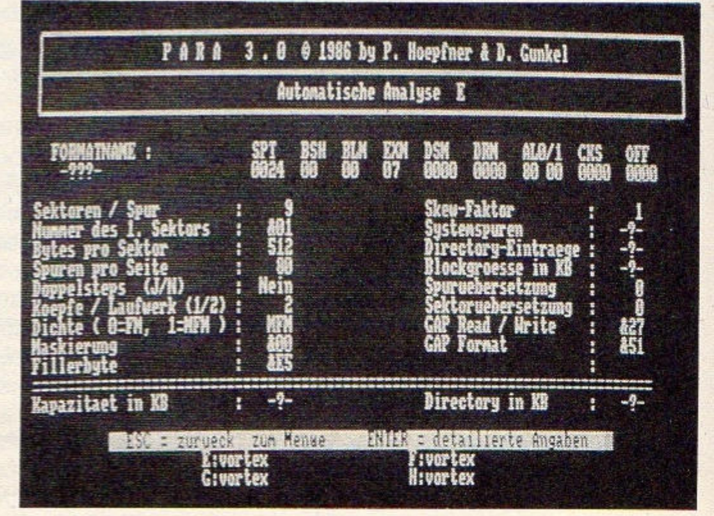
Etwas anders verhält es sich mit Diskpara von Strauß Elektronik. Da Diskpara direkt auf Disk-Controller-Routinen zugreift, ist es nur unter dem Original-CP/M 2.2 lauffähig. Jeder Eingriff in die Zeropage oder die TPA verändert das CP/M so, daß die Adressen verschoben sind. Dies trifft also auf CP/M 2.2 zu, sobald unter der Vortex- oder dk'tronics-Speichererweiterung eine 62-KByte-TPA eingerichtet wird. Der Funktion von Diskpara tut das keinen Abbruch, lediglich die von Vortex verschobenen Adressen zur Einbindung der Speichererweiterung sind nicht ansprechbar, also keine TPA-Vergrößerung unter Diskpara. Nach dem Gebrauch von Diskpara kann man ja wieder das große CP/M booten. Das eingestellte Format auf dem B-Laufwerk ist zwar anschließend verschwunden, aber die Daten von den unterschiedlichsten Formaten lassen sich problemlos auf das A-Laufwerk übertragen.

```
Doublestep              :  N
Anzahl der Sektoren     :  9
Phys. Sektorgroesse     :  512
Erster Sektor           :  1
Letzter Sektor          :  9
Hardware Skew           :  2

Reihenfolge der Sektoren auf der Diskette :
Vorderseite :  1  6  2  7  3  8  4  9  5

Soll die Rueckseite auch analysiert werden ?
Bei einer einseitigen Diskette wird ein Kaltstart noetig
```

Diskpara zeigt bei der Analyse nur wenig Details



Para 3.0 geizt nicht mit Informationen

der Rest der CP/M-Welt

Alle Standard-Programme unter CP/M 2.2, die nur mit den Standard-BDOS-Routinen ohne BIOS-Aufrufe arbeiten, laufen uneingeschränkt. Dazu gehören die meisten Public-Domain-Utilities (zum Beispiel Nsweep, Du) und auch professionelle Programme wie Wordstar. Endlich hat man für Texte genügend Platz auf der Diskette, ohne daß die häufige Meldung »Disc full« erscheint. In der neuesten Diskpara-Version kann man auch das MS-DOS-Format direkt einstellen.

Die CP/M-2.2-Version für den CPC 664 läuft auch auf dem CPC 6128. Für diesen bietet sich eine ideale Kombination an: Diskpara und die dk'tronics Silicon-Disk. (Eine zusätzliche Speichererweiterung, die man genau wie ein Diskettenlaufwerk anspricht.) CP/M Plus kennt einen erweiterten BIOS-Teil, das sogenannte XBIOS, in dem die Diskparameter einstellbar sind. Da die Silicon-Disk sich streng am Amstrad-CP/M Plus orientiert, gibt es keine Konflikte mit Diskpara. Mit der Silicon-Disk als Zwischenspeicher lassen sich jederzeit die Formate wechseln und so zwischen beliebigen CP/M-Disketten Daten austauschen. Auch nach einem Reset (<CTRL+SHIFT+ESC>) und Warmstart im CP/M 2.2 ist der Zugriff auf die unveränderten Dateien der Silicon-Disk gewährleistet. Ist Diskpara unter CP/M-Plus aktiv, läßt sich auch ohne Originaldiskette ständig das Format im Laufwerk B ändern. Unter der CP/M-2.2-Version gilt das zwar grundsätzlich auch, man darf lediglich nicht vergessen, die Disketten mit <Ctrl-C> einzuloggen.

Bei der CP/M-Plus-Version sind alle Programme uneingeschränkt nutzbar, auch dBase II und Multiplan beispielsweise.

Die Anwendung von Diskpara ist denkbar einfach und so gut menügesteuert, daß man selbst ohne Anleitung durch das Programm findet. Hilfe erhält vor allem der weniger Geübte bei der Eingabe von Disketten-Parametern, indem zu jedem Änderungspunkt in einem Window am unteren Bildrand Erläuterungen über sinnvolle Bereiche gegeben werden. Wer das Programm kennt und die Nummer des gewünschten Formats weiß, kann mit dem Aufruf »Diskpara 27« zum Beispiel das Vortex-Format einloggen. Dabei werden nur die notwendigen Teile des Programms durchlaufen und die Einstellzeit verkürzt sich auf ein Minimum.

Der günstige Preis und die volle CP/M-Kompatibilität machen Diskpara zu einem nützlichen Werkzeug für den Austausch mit anderen Systemen oder zur besseren Nutzung des Diskettenplatzes. Diskpara kann man übrigens auch auf 40-Spur-Laufwerken einsetzen, solange man als Spur-Anzahl entsprechende Werte wählt. Empfehlenswert ist das Paket-Angebot für etwa 98 Mark, bestehend aus Diskpara und MS-Copy; einem Kopier-Programm, das den Daten-Transfer zwischen CP/M und MS-DOS Dateien übernimmt. Das Format von MS-Copy ist so gewählt, daß es von einem 40-Spur-Laufwerk sofort ohne Diskpara lesen kann. MS-Copy ist auch einzeln zu beziehen, dann allerdings um einiges teurer.

Fazit: Diskpara hat zwar weniger Funktionen als Para 3.0, ist aber rasch erlernbar und leicht zu bedienen. Das Standard-CP/M wird von Diskpara unterstützt. Vor allem für CPC-6128-Besitzer mit (oder ohne) dk'tronics Silicon-Disc ein unabdingbares Muß. Vorausgesetzt wird jedoch ein Zweitlaufwerk mit 5¼- oder 3½-Zoll-Disketten, am besten mit 80 Spuren. Das 62-KByte-CP/M-2.2 ist bis jetzt noch nicht gleichzeitig verwendbar, es wird aber daran gearbeitet. Keinerlei Einschränkungen gibt es dagegen unter CP/M-Plus.

Für VDOS-Anwender ist dagegen Para 3.0 ein Volltreffer. Es unterstützt nicht nur die Vortex-Hardware (VDOS-Controller und Vortex-Laufwerk), sondern sie sind Voraussetzung für die Anwendung von Para 3.0. Trotz des höheren Preises ist Para 3.0 ein umfangreiches Dienstprogramm, das vor allem dem fortgeschrittenen Anwender den Daten-Austausch mit fast beliebigen anderen Computersystemen erleichtert, sowie eine genaue und automatische Analyse unbekannter Formate bietet.

(Helmut Jungkunz/kl)

Die wichtigsten Eigenschaften auf einen Blick

Programmname: Para 3.0	Diskpara
Hersteller: Vortex GmbH	Frank Strauß Elektronik
Lieferumfang: Handbuch, 5¼-Zoll-Diskette	Handbuch 5¼- oder 3-Zoll-Diskette
Hardware: alle CPC-Modelle mindestens ein Vortex-Laufwerk (auch X-Laufwerke)	CPC 464 mit CP/M 2.2 oder CPC 6128 mit CP/M (auch Plus) ein Zweitlaufwerk (80-Spur)
Preis: 149 Mark	79 Mark
Zusätze: Copy-Dos (zus. 198 Mark) (MS-DOS-Dateitransfer)	MS-Copy (zus. 99 Mark) (MS-DOS-Dateitransfer)
Vorteile: - erlaubt eigene Formatbibliotheken - BIOS-Manipulationen - komfortable Analyse - residente Einbindung logischer Laufwerke - eingebautes Kopier-Programm - einzelnes Format ohne RAMBIOS-Aufruf nicht einstellbar - Kopierschutz sieht zwei Sicherheitskopien vor (Auto-Backup)	- 120 CP/M-Formate einstellbar - CP/M 2.2 mit kleiner TPA genügt - voll kompatibel - CP/M Plus-kompatibel - dk'tronics Silicon-Disc nutzbar - gute, eindeutige Menüführung - sehr rasch einzelnes Format einstellbar - gutes Preis-Leistungs-Verhältnis
Nachteile: - bei RAM-BIOS-Aufruf ist TPA 10 KByte kleiner - nicht kompatibel zum Standard-CP/M 2.2 - einzelnes Format ohne RAM-BIOS-Aufruf nicht einstellbar - kein CP/M-Neustart mit X-Modul ohne Reduktion der TPA - relativ hoher Preis	- Kopierschutz erfordert stets die Original-Diskette bei Kaltstart - Manipulationen der TPA (Vortex-RAM-Disk) und großes CP/M bisher nicht unterstützt (folgt später)

Sprachen auf einen Blick

Wer sich mit sprachlichen Alternativen zum eingebauten Locomotive Basic seines CPC auseinandersetzen will, sieht sich einem schier unüberschaubaren Angebot gegenüber. Mit unserer großen Marktübersicht wollen wir helfen, Licht ins Dunkel zu bringen.

Immer wieder hört man als Computer-Hobbyist, was die Programmiersprache X für wunderbare Leistungen verspricht, welche Geschwindigkeitsvorteile der Dialekt Y bietet oder daß gar der Compiler Z die Sprache der Zukunft sei. Wer kann sie nicht nachfühlen, diese Unsicherheit, welche Sprache nun für den heimischen Computer erhältlich ist, welche konkreten Compiler beziehungsweise Interpreter es bereits gibt und nicht zuletzt, wo sie zu bekommen sind. Gerade für die CPC-Serie ist die Vielfalt der Produkte am Markt für den einzelnen nicht zu überblicken. Hier finden Sie nun eine Übersicht der wichtigsten Programmiersprachen und erfahren, wo Sie nähere Informationen darüber erhalten. Damit steht aufregenden neuen Perspektiven nichts mehr im Weg. (kl/ja)

Programmname	464	664	6128	Wieviel Speichererweiterung für 464 und 664	CP/M	CP/M Plus	51/4-Zoll-Diskette	3-Zoll-Diskette	Kassette	Modul	Compiler	Interpreter	Anleitung deut./engl.	Preis	Anbiet.	Besonderheiten
Basic																
Basic-Compiler	x			–	x			x	x		x		d	49,95	RU	auf Diskette 69,75 Mark
CBasic	x	x	x	–	x			x	x		x		d	174,–	MT/SD	bedient GSX
Laser Basic	x	x	x	–				x	x		x		d/e	69,95	DS/RU	auf Diskette 79,95 Mark
MS Basic	x	x	x	–	x	x		x			x	x	d	199,–	MT	mit Makroassembler u. Linker
Nevada Basic	x	x	x	–	x	x	x	x				x	e	89,–	TC	Matrizenverarbeitung; Fullscreen-Editor
XBC-Basic	x	x	x	–				x	x		x		d	99,–	BG	mit Fließkomma-Arithmetik; Grafik- u. Soundunterstützung lieferbar ab August 1987
Turbo Basic	x	x	x	–				x	x		x		e	79,85	DS	auf Diskette 109,85 Mark
Assembler																
Assembler/Disassembler	x	x	x	–				x	x			x	d	129,–	SR	auf Diskette 145,– Mark
Assembler-Kurs	x	x	x	–				x					d	75,–	SY	kompletter Assembler eingebaut
CPC-Macro	x	x	x	–	x	x		x	x				d	98,–	HO	auf Diskette 139,– Mark; Treiber für CP/M-Anpassung
Deep Thought	x	x	x	–					x				e	33,–	DC	Two Pass Assembler Bildschirmeditor
Dev Pac	x			–					x				e	69,–	DS	Assembler/Disassembler
GMON 1.3	x	x	x	–	x			x	x					59,–	GO	auf Diskette 89,– Mark Maschinensprache-Monitor mit Assembler, Disassembler; unterstützt in d. Diskettenversion Bankumschaltung; neben Amsdos-Version wird eine CP/M-Version mitgel.
Kassemble 12	x	x	x	–				x			x		d	98,–	DA	Linkfunktion; Compilieren von Diskette aus mehreren Sourcecode-Dateien bis zu 12 KByte Objektcode; sehr schnell (150 KByte ca. 3 min)
Laser Genius	x	x	x	–				x	x				d/e	59,85	DS	auf Diskette 79,85 Mark; Makrosprache Phoenix; Analyzer; Debugger
Maxam	x	x	x	–				x	x	x			e	69,85	DS	auf Diskette 89,90 Mark; auf ROM 128,85 Mark; Assembler, Disassembler
Maxam II	x	x	x	–		x		x					e	278,–	DS	Makroassembler mit Linker
Philosoft Assembler	x	x	x	–				x		x			d		PH	Wordstar-kompatibler Editor
Profimat CPC	x	x	x	–				x					d	99,–	DB	Assembler m. Monitor; Einzelschrittmodus; Verketten von langen Quelltexten möglich
Pyradev Assembler	x	x	x	–				x					e	99,–	RU	Toolkit
Star-Mon	x	x	x	–				x	x				d	59,90	ST	auf Diskette 79,90 Mark; läuft auch unter CP/M; mit Editor, Monitor, Diskmonitor; Trace-Funktion
Super Pac 80	x	x	x	–				x	x				d	ca.130,–	PS	komplettes Z80-Entwicklungssystem; Singlestep; auf Disk. ca. 140,– Mark
Z-80 Assembler Paket	x	x	x	–	x	x	x	x					d	30,–	MK	Public Domain; Linker, intelligenter Disassembler, Monitor

Programmname	464	664	6128	Wieviel Speichererweiterung für 464 und 664	CP/M	CP/M Plus	5 1/4-Zoll-Diskette	3-Zoll-Diskette	Kassette	Modul	Compiler	Interpreter	Anleitung deut./engl.	Preis	Anbiet.	Besonderheiten
diverse Sprachen																
Arnor-C	x	x	x	64 KByte dk'tronics		x		x			x		e	auf Anfrage	AR	–
Aztec-C	x	x	x	64 KByte	x	x		x			x		e	639,–	BS	Version mit Sourcecode der C-Library kostet 1122,– Mark
BCPL	x	x	x	–				x			x		e	128,85	DS	Lieferung mit ROM; läuft auch unter CP/M und Amsdos; Texteditor und Spiel-Sourcecode
C-80	x	x	x	–	x	x	x	x			x		e	189,–	TC	konfigurierbar mit vielen Assemblern
C ++	x	x	x	–	x	x		x			x		e	178,85	DS	läuft nicht mit Vortex-Speichererweiterung; erweitertes GSX
C-Package	x	x	x	–		x		x			x		e	278,–	DS	Fließkomma-Arithmetik; 16- und 32-Bit-Arithmetik
Comal	x	x	x	–				x		x		x	d	69,–	CO	auf ROM 248,– Mark
CPC-Forth	x	x	x	–	x	x		x	x		x	x	d	98,–	HO	auf Diskette 139,– Mark; FIG-Standard, Screeneditor, UNDO-Funktion, RAM-Disk
Digital Research Pascal MT+	x	x	x	64 KByte	x	x		x			x		e	174,–	MT/SD	–
Fig Forth	x	x		–					x			x	e	98,95	DS	–
Forth-83	x	x	x	–			x	x	x		x	x	d	148,–	FS	auf Diskette 178,– Mark; mit Trace-Funktion
Forth-83	x	x	x	–	x	x	x	x			x	x	d	30,–	MK	Public Domain; es ist nicht identisch mit Forth-83 von FS; Bildschirmeditor, Inline-assembler, Decompiler
Hisoft C-Compiler	x	x	x	–	x	x		x			x		e	189,–	SD/DS	läuft auch unter Amsdos
Hisoft Pascal	x	x	x	–				x	x		x		e/d	199,–	RU	auf Diskette 215,– Mark
JRT-Pascal	x	x	x	64 KByte	x	x	x	x			x		d	30,–	MK	Public Domain; Editor, Linker und 8080-Assembler
Künstliche Intelligenz	x	x	x	64 KByte	x	x	x	x				x	d	30,–	MK	Public Domain; XLISP- und E-PROLOG-Interpreter
Lisp/80	x	x	x	–	x	x	x	x				x	e	159,–	TC	an Interlisp angelehnt
Logo	x			–					x				d	ca. 49,–	DS	–
MI-C	x	x	x	64 KByte	x	x		x			x		d	445,–	HR	Laufzeitüberwachung; Inline-assembler
Morrison Pascal	x	x	x	–						x	x		e	98,85	DS	Mischung aus Turbo-Pascal und Basic; Standalone-Programme sehr kurz
Nevada Cobol	x	x	x	–	x	x	x	x			x		e	99,– b. 189,–	TC/SD	–
Nevada Fortran	x	x	x	–	x	x	x	x			x		e	99,– b. 189,–	TC/SD	–
Nevada Pascal	x	x	x	–	x	x	x	x			x		e	89,–	TC	BCD-Arithmetik
Nevada Pilot	x	x	x	–	x	x	x	x			x		e	89,–	TC	–
Oxford Pascal								x			x		e	109,–	RU	–
Pascal 80	x	x	x	–	x	x		x			x		e	178,85	DS	läuft nicht mit Vortex-Speichererweiterung; erweitertes GSX
Small-C	x	x	x	64 KByte	x	x		x			x		d	99,–	MT	mit Small-Mac-Assembler, Small-Tolls in C-Sourcecode
Small-C	x	x	x	64 KByte	x	x	x	x			x		d	30,–	MK	Public Domain; es ist nicht identisch mit Small-C von MT; Fließkomma-Arithmetik, Inlineassembler
Toolworks C-80	x	x	x	64 KByte	x	x		x			x		e	180,–	BS	Erweiterungsmodul für Fließkomma-Arithmetik: 104 Mark
Turbo Pascal 3.0 3.0	x	x	x	–	x			x						225,72	BR/MT/DA/HS	–
Turbo Pascal 3.0 mit Grafik-Toolbox	x	x	x	–	x			x						285,–	HS/MT/DA	-

Weitere Informationen erhalten Sie bei:
AR: Arnor Deutschland, H.-H.-Jahn-Weg 21, 2000 Hamburg 76
BG: BBG-Software, Beimoorweg 2-4, 2070 Ahrensburg
BS: BSP Thomas Krug, Weißenburgstr. 49, 8400 Regensburg
CO: Comalvertrieb D. Belz, 2270 Utersum
DA: data berger, Im Lichtenfelde 76, 4760 Paderborn
DB: Data Becker, Merowingerstr. 30, 4000 Düsseldorf 1
DC: Deltacom, Postfach 52019, 4600 Dortmund 50
DS: Denisoft, Postfach 106421, 2800 Bremen
FS: Forthsysteme A. Flesch, Postfach 1103, 7814 Breisach
GO: U. Goedan, K-Furterstr. 46, 7408 Kusterdinger
HO: Holtkötter, Hallerplatz 15, 2000 Hamburg 13
HR: Herbert Rose EDV, Bogenstr. 32, 4390 Gladbeck
HS: Heimsoeth, Fraunhoferstr. 13, 8000 München 5
MK: Martin Kotulla, Grabbestr. 9, 8500 Nürnberg 90
MT: Markt&Technik, Hans-Pinsel-Str. 2, 8013 Haar
PH: Philosoft, Pariser Platz , 8000 München 80
PS: Profisoft, Sutthauser Str. 50/52, 4500 Osnabrück
RU: Rushware, An der Gümpgesbrücke 24, 4044 Kaarst 2
SD: Schneider Data, Rindermarkt 8, 8050 Freising
SR: Schneider Rundfunkwerke, Postfach 120, 8939 Türkheim
ST: Star Division, Zum Elfenbruch 1, 2120 Lünburg
SY: Sybex-Verlag, Vogelsanger Weg 111, 4000 Düsseldorf 30
TC: Tesco, Postfach 10, 8714 Wiesenheid

Noch mehr Eingabekomfort

Hier wieder der Checksummer für den Schneider CPC! »Explora 2.0« macht die Eingabe von Programmen ganz einfach.

Zuerst einmal Informationen für alle, die noch nicht wissen, was »Explora« ist: Wenn Sie dieses Programm gestartet und wieder gelöscht haben, überprüft der Computer automatisch Ihre Eingaben auf Richtigkeit. Sobald Sie die Eingabe einer Programmzeile abschließen, erscheint eine vierstellige Hexadezimalzahl in eckigen Klammern auf dem Bildschirm. Das im Heft abgedruckte Listing enthält ebenfalls solche Zahlen. Stimmen die Prüfsummen auf dem Bildschirm und im Heft überein, haben Sie die Zeile korrekt abgetippt. Gibt es Unterschiede zwischen den Werten, sollten Sie auf Fehlersuche gehen und die Zeile korrigieren. Das alles konnte »Explora 1.0« auch schon. Der Vorteil der neuen Version besteht darin, daß Sie jetzt größere Freiheit bei der Eingabe der Zeilen haben. So akzeptiert unser Prüfsummenprogramm die Basic-Schlüsselworte in Klein- oder Großbuchstaben (auch gemischt). »PRINT« läßt sich mit dem Fragezeichen abkürzen. »Explora 2.0« läßt zum Beispiel für die Zeile »100 PRINT« folgende Eingaben zu:

```
100 PRINT
100 print
100 ?
100 PrInT
```

Die Zeilen müssen also nicht mehr schon beim Eintippen so aussehen wie im Heft, sondern erst beim Auflisten. Außerdem werden Prüfsummen nur noch für Programmzeilen ausgegeben, nicht mehr – wie früher – auch bei Direktbefehlen. Vor der Zeilennummer stehende Leerzeichen, Line-Feeds und Tabulatorzeichen überliest Explora jetzt selbsttätig. Leerzeichen innerhalb der Zeile wertet es aber weiterhin. Sie verändern also die Prüfsumme. Explora erlaubt auch die Verwendung des EDIT-Befehls. AUTO ist jetzt ohne Einschränkungen zu benutzen – allerdings nur beim CPC 664 und CPC 6128. Explora 1.0 liegt im Speicher fest zwischen den Adressen A000 und A086 hex. Die neue Version verschiebt der Basic-Lader automatisch im Speicher direkt unter HIMEM. So ist SYMBOL AFTER einwandfrei funktionsfähig. Eine kleine Einschränkung gibt es aber doch: Löschen Sie keinesfalls Zeilen durch Eingabe der Zeilennummer und anschließendes Drücken der ENTER-Taste! Die Zeile wird nämlich gar nicht wirklich gelöscht, sondern erscheint als Duplikat der folgenden Zeile. Verwenden Sie statt dessen DELETE. Statt »20« schreiben Sie »DELETE 20«. Das Wichtigste nicht zu vergessen: Explora 2.0 ist aufwärtskompatibel zur Version 1.0. Das heißt, daß Sie sowohl mit Explora 2.0 frühere Listings abtippen können, als auch mit Explora 1.0 alle zukünftigen. Die Prüfsummen sind identisch.

Aber bei den gedruckten Listings hat sich einiges geändert. Die Neuerungen betreffen die Darstellung von Leer- und Sonderzeichen. Statt "{5 SPACE}" steht jetzt im Listing "<u><5></u>" für fünf Leerzeichen. Um dies eindeutig vom tatsächlichen Programmcode zu unterscheiden, erscheint der Text unterstrichen. Die Steuerzeichen hießen bisher beispielsweise »{CTRL A}«. Jetzt steht hier die übersichtlichere Form <u>A</u>. Finden Sie im Listing also einen unterstrichenen Buchstaben ohne Klammern, müssen Sie gleichzeitig die CTRL-Taste drücken. Grafikzeichen stehen zukünftig in Klammern und sind als ASCII-Wert mit vorangehendem »G« für »Grafikzeichen« dargestellt. Das Zeichen 223 hat dann im Listing die Form <G223>. Die Zeichen können nicht von der Tastatur aus direkt eingegeben werden. Simpler Trick: Ausgabe des Zeichens mit »PRINT CHR$(223)« und Übernahme mit dem Copy-Cursor.

Sämtliche Listings sind im ASCII-Zeichensatz gedruckt. Deutsche Sonderzeichen erscheinen daher im Druck als Klammern und andere amerikanische Zeichen. Verwenden Sie ruhig anstelle dieser Zeichen die entsprechenden deutschen.

(Martin Kotulla/ja)

```
100 ' ***************************          [DFCC]
110 ' *                         *          [FADA]
120 ' *       EXPLORA V2.0      *          [761E]
130 ' *                         *          [DCDE]
140 ' ***************************          [C3D4]
150 '                                      [E1BA]
160 DEF FNlsb(x)=255 AND UNT(x)            [39E0]
170 DEF FNmsb(x)=255 AND INT(x/256)        [8864]
180 SYMBOL AFTER 256:MEMORY HIMEM-161      [948C]
190 start=HIMEM+1:SYMBOL AFTER 240         [2092]
200 FOR i=&A000 TO &A09D:READ a$:sum=sum
    +VAL("&"+a$):NEXT i                    [B2C8]
210 IF sum<>19814 THEN PRINT "DATA-Fehle
    r!":END                                [FCCE]
220 RESTORE:FOR i=start TO start+&9D:REA
    D a$                                   [608E]
230 POKE i,VAL("&"+a$):NEXT i              [24D2]
240 FOR i=1 TO 5:READ a:a=a+start          [AC2A]
250 wert=PEEK(a)+PEEK(a+1)*256-40960+sta
    rt                                     [2776]
260 POKE a,FNlsb(wert):POKE a+1,FNmsb(we
    rt):NEXT i                             [01B2]
270 IF PEEK(6)=&80 THEN ed=&BD3A:POKE &B
    F20,&A4                                [56A8]
280 IF PEEK(6)=&7B THEN ed=&BD5B:POKE &B
    F20,&8A:RESTORE 470                    [760C]
290 IF PEEK(6)=&91 THEN ed=&BD5E:POKE &B
    F20,&8A:RESTORE 490                    [16FA]
300 POKE &BF21,&AC:POKE &BF22,PEEK(ed)     [71DE]
310 POKE &BF23,PEEK(ed+1):POKE &BF24,PEE
    K(ed+2)                                [9984]
320 POKE ed,&C3:POKE ed+1,FNlsb(start):P
    OKE ed+2,FNmsb(start)                  [9AE6]
330 IF PEEK(6)=&80 THEN END                [6044]
340 FOR i=1 TO 7:READ a$,b$:a=VAL("&"+a$
    )+start:b=VAL("&"+b$)                  [3306]
350 POKE a,FNlsb(b):POKE a+1,FNmsb(b):NE
    XT i                                   [0332]
360 DATA CD,22,BF,F5,C5,D5,E5,2A,20,BF,C
    D,61,DD,B7,28,62                       [5BFC]
370 DATA E5,2A,20,BF,CD,88,A0,E1,30,58,C
    D,04,EE,CD,A3,E7                       [5EF2]
380 DATA CD,63,E1,ED,4B,20,BF,21,00,00,0
    A,5F,16,00,19,03                       [DBF6]
390 DATA FE,00,20,F6,DD,2A,20,BF,01,00,0
    0,DD,7E,00,5F,16                       [4D3E]
400 DATA 00,19,04,F5,A8,47,F1,09,DD,23,F
    E,00,20,ED,3E,0D                       [E53C]
410 DATA CD,5A,BB,3E,0A,CD,5A,BB,3E,5B,C
    D,5A,BB,7C,CD,77                       [259A]
420 DATA A0,7C,CD,7B,A0,7D,CD,77,A0,7D,C
    D,7B,A0,3E,5D,CD                       [014A]
430 DATA 5A,BB,E1,D1,C1,F1,C9,1F,1F,1F,1
    F,E6,0F,C6,30,FE                       [A10A]
440 DATA 3A,38,02,C6,07,C3,5A,BB,CD,61,D
    D,B7,37,C8,CD,04                       [64AC]
450 DATA EE,D0,7E,FE,20,20,01,23,CD,D2,E
    6,37,9F,C9                             [0C36]
460 DATA &15,&5F,&63,&67,&6B               [3A22]
470 DATA 0B,DE52,1B,EED4,1E,E869           [7B14]
480 DATA 21,E259,89,DE52,8F,EED4,99,E7AA
                                           [0586]
490 DATA 0B,DE4D,1B,EECF,1E,E864           [1F52]
500 DATA 21,E254,89,DE4D,8F,EECF,99,E7A5
                                           [249A]
510 END                                    [AA1A]
```

Listing. »Explora« macht Eingabefehler fast unmöglich

Steckbrief	
Programm:	Explora 2.0
Computer:	CPC 464/664/6128
Checksummer:	Explora 1.0
Datenträger:	Kassette, Diskette

Nie mehr DATAs mit »CPC«

Oft bestehen leistungsfähige Programme zum Teil aus Maschinencode. Leider bedingte dieser Umstand bisher, daß Sie ellenlange »Data-Wüsten« als Listing eingeben mußten. »CPC« macht damit nun ein für allemal Schluß.

Zur Eingabe langer Maschinencode-Programme ist nichts empfehlenswerter als das Programm »CPC«. Der Name kommt nicht von ungefähr – er steht für »Comfortable Program for Codeinput«. Diesen Komfort gewährleisten die vielfältigen Fähigkeiten des CPC. Er ist praktisch an beliebige Maschinencode-Listings anzupassen: Weder vor DATA-Ladern noch vor Hexdumps streckt »CPC« seine Waffen. Einzige Bedingung für Basic-Lader mit Prüfsummen ist, daß die Zahl der Byte pro Zeile konstant bleibt. Für viele Leser brauchen wir die Details sicher nicht zu wiederholen, da sie bereits aus einem der letzten Schneider-Sonderhefte bekannt sind. Für neu hinzugekommene Leser führen wir sie hier dennoch auf. Die Vorzüge der neuesten Version (identisch mit der aus Sonderausgabe 13):

- Dateinamen darf man jetzt in beliebigem Format eingeben. Wer mit Kassetten arbeitet, darf also Namen mit einer Länge von bis zu 16 Zeichen verwenden, wer Disketten bevorzugt, setzt wahlweise die Laufwerksnummer oder den User-Bereich voran.
- »CPC« setzt HIMEM automatisch auf Adresse 20000.
- Der Aufruf der Routine »Parameter einstellen« erfolgt automatisch bei der Code-Eingabe oder DATA-Erzeugung.
- Das Unterprogramm »Parametereingabe« arbeitet komfortabler und ist dadurch einfacher einzustellen. Die einzelnen Punkte sind numeriert. Ein Druck der jeweiligen Zifferntaste wechselt die Einstellungen (zum Beispiel von »hex« auf »dez«), wie Sie im Bild sehen können.
- Die Routine »Erzeuge DATAs« verarbeitet jetzt auch die Startadresse 8000 hex korrekt und die Vorgabe eines Offset ist berichtigt.
- Der erzeugte Basic-Lader ist kürzer und schneller und erhält automatisch den SAVE-Befehl zur Speicherung des erzeugten Maschinencodes.
- Jetzt sind auch DATA-Lader ohne Prüfsumme oder mit Add- statt mit Hash-Prüfsumme zu erzeugen.
- Durch diverse Einsparungen hat »CPC« nur noch eine Länge von 10 KByte.
- Zwei Hilfsprogramme unterstützen das Hauptprogramm »CPC.BAS« (Listing 1). Für noch komfortablere Bedienung belegt »CPC.HLP« (Listing 2) die Funktionstasten mit allen Hex-Ziffern und setzt die Farben auf eine augenfreundliche Kombination. »CPC.INF« finden Sie nur auf der Leserservice-Diskette, denn es enthält eine Kurzanleitung für »CPC«.
- Die Routine »Code eingeben« verarbeitet jetzt bis zu 128 Byte formatierter Eingabe. So sind nun auch DATA-Lader mit sehr langen Zeilen abzutippen.
- Die Eingabe von Dezimalzahlen war bisher etwas kompliziert. Um das nötige dreistellige Format einzuhalten, waren bei ein- oder zweistelligen Zahlen entweder führende Nullen voranzuschicken oder stets die Leertaste (oder <ENTER>) zu drücken. Nun ist, wenn Sie im Parameter-Menü »Ende annehmen« auf »Nein« einstellen, jedes Byte mit der Leertaste oder <ENTER>, beziehungsweise dem Dezimalpunkt zu bestätigen.

Das Programm »CPC« verhilft in komfortabler Weise zu einer einfachen, schnellen und sicheren Eingabe von Maschinencode-Programmen. »CPC«-Benutzer geben nur zirka 60 Prozent des Listingumfangs von Basic-Ladern ein.

Nach dem Start erscheint das Hauptmenü mit fünf Punkten. Die Eingabe der Anfangsbuchstaben ruft das jeweilige Unterprogramm auf.

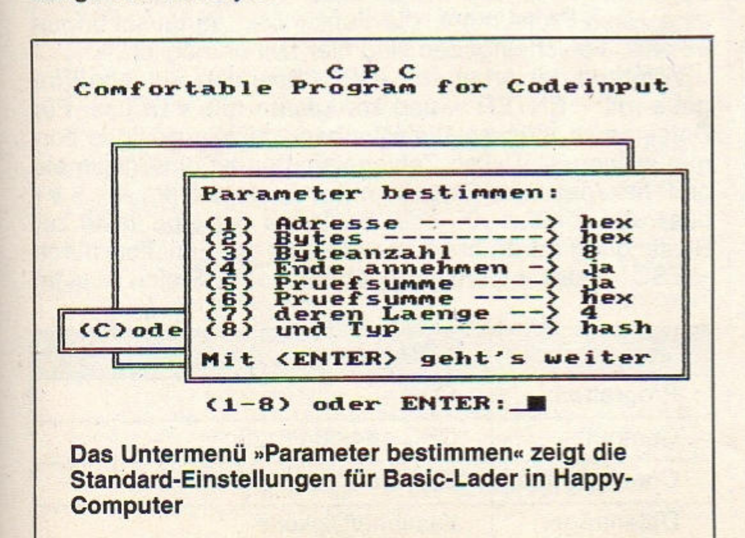
Lade Code

lädt eine Binärdatei von Kassette oder Diskette. Die Ladeadresse des Programms müssen Sie eingeben; sie darf jedoch nicht unter 20000 liegen. Andernfalls laden Sie es an eine höhere Adresse. Drücken Sie bei der Aufforderung, den Datenträger bereitzumachen, <ESC> oder <CTRL+C>, bricht die Routine ab.

Schreibe Code

sichert den Inhalt eines Speicherbereichs auf Diskette oder Kassette. Sie müssen die Anfangs- und die Endadresse angeben. Beide Werte lassen sich an den Zeilenadressen der DATA-Listings ablesen. Haben Sie den Code in einen anderen Bereich geladen oder eingegeben, weil es sonst unterhalb der Adresse 20000 stünde, berücksichtigt CPC das hier nicht. Sie müssen später den Code an die richtige Adresse laden (zum Beispiel: »LOAD "CODE",3000«). Auch diese Routine ist mit <ESC> oder <CTRL+C> abzubrechen.

Code eingeben



Das Untermenü »Parameter bestimmen« zeigt die Standard-Einstellungen für Basic-Lader in Happy-Computer

```
1000 '--> CPC.HLP -- Voreinstellungen        [A3EC]
1010 '                                       [9212]
1020 '--> Farben                             [EC20]
1030 '                                       [9016]
1040 BORDER 4:INK 0,4:INK 1,26:INK 3,18      [0A90]
1050 '                                       [B61A]
1060 '--> Tastatur - Zehnerblock             [B004]
1070 '                                       [B41E]
1080 KEY DEF 15,1,48,127,128                 [A192]
1090 KEY DEF 13,1,49,47,129                  [B936]
1100 KEY DEF 14,1,50,47,130                  [7408]
1110 KEY DEF 5,1,51,47,131                   [2EAE]
1120 KEY DEF 20,1,52,65,132                  [150E]
1130 KEY DEF 12,1,53,66,133                  [C718]
1140 KEY DEF 4,1,54,67,134                   [7CC2]
1150 KEY DEF 10,1,55,68,135                  [CD24]
1160 KEY DEF 11,1,56,69,136                  [9C2E]
1170 KEY DEF 3,1,57,70,137                   [81C6]
1180 KEY DEF 7,1,46,38,138                   [6DD6]
1190 KEY DEF 6,0,13,139,140                  [D01E]
1200 '                                       [9514]
1210 'Version vom 22.10.86                   [BA24]
```

Listing 1. So ist der Zehnerblock hilfreich belegt

ist die wichtigste Routine des »CPC«. Zunächst geben Sie im automatisch erscheinenden Unter-Menü »Parameter einstellen« die für das Listing erforderlichen Standards ein. Darauf folgt die Vorgabe der ersten Zeilennummer und der Schrittweite.

Drücken Sie hier einfach <ENTER>, bleibt der Zeilenzähler auf Null. Nun geben Sie noch die Startadresse vor. Bei der Eingabe der Byte-Werte ist die Gefahr von Fehleingaben sehr gering, da nur die Ziffern 0 bis 9, beziehungsweise die Buchstaben A bis F zugelassen sind. Ein Druck der Leertaste, <ENTER> oder des Punktes formatiert alle führenden Nullen automatisch und schließt die Eingabe eines Byte ab. Die Tasten mit Schrägstrichen (</> und <\>) wiederholen das zuletzt eingegebene Byte. Dadurch ist gewährleistet, daß Sie nur ein Minimum einzugeben haben. Verzichten Sie auf Prüfsummen (wovon wir aber eindringlich abraten, denn etwaige Eingabefehler sind so kaum zu finden), ist damit die Eingabe beendet. Ansonsten geben Sie nun die Prüfsummen ein. Ist die Prüfsumme korrekt, ertönt ein Signalton und Sie gehen zur nächsten Zeile. Fehler korrigieren Sie mit Hilfe der DEL-Taste.

```
100 '************************************  [31D4]
101 '*BEISPIEL.DAT - DATA-Lader von 'CPC'*  [1BA8]
102 '************************************  [A3D8]
103 '                                       [DEB6]
104 DATA 9C40,01,0A,A0,21,0E,A0,CD,D1,17AB  [489C]
105 DATA 9C48,BC,C9,12,A0,18,0A,00,00,64E8  [3F74]
106 DATA 9C50,00,00,53,43,41,4C,C5,00,0CE2  [0616]
107 '                                       [DEBE]
108 '                                       [FEC0]
109 '                                       [FEC2]
110 '                                       [DDB2]
126 DATA 9CF0,19,22,A8,AC,22,2C,B3,2A,1B2C  [CBCA]
127 DATA 9CF8,AA,AC,22,2E,B3,E1,23,C1,7E3B  [3820]
128 DATA 9D00,10,93,FB,C9,00,00,00,00,3F30  [A316]
129 DATA *ENDE*                            [83CC]
130 adr=&9C40:zeile=104:MEMORY adr-1        [EA36]
131 READ d$:IF d$="*ENDE*"THEN 142          [0182]
132 '                                       [DFBA]
133 '                                       [DFBC]
134 '                                       [FFBE]
135 '                                       [FFC0]
140 IF pr<>pr2 THEN PRINT"Pruefsummenfehler
     in Zeile";zeile:STOP                   [3E0A]
141 zeile=zeile+1:GOTO 131                  [B052]
142 SAVE"BEISPIEL.BIN",B,&9C40,&C7:END      [0AF6]
```

Ein beispielhafter Ausschnitt eines typischen DATA-Laders. Die Ziffern im hell unterlegten Bereich sind in jedem Fall einzugeben, während Sie auf die Eingabe der Prüfsummen (dunkel unterlegt) verzichten können. Dazu kommen noch die Startadresse (in diesem Fall 9C40 hex) und zum Speichern die Endadresse (hier 9D03 hex). Den Rest des Basic-Laders ersparen Sie sich mit »CPC«.

Erzeuge DATAs

erzeugt aus Maschinencode im Arbeitsspeicher einen lauffähigen Basic-Lader auf Diskette. Gehen Sie bitte wie folgt vor: Stellen Sie zuerst die korrekten Parameter ein. Wählen Sie dann die Namen für den DATA-Lader und die später vom Lader zu erzeugende Binär-Datei. Nun erwartet »CPC« die Anfangs- und die Endadresse des Maschinencodes. Da dieser nicht unter 20000 beginnen darf, manche Programme aber nur auf niedrigeren Adressen arbeiten, können Sie hier einen Offset von der Ladeadresse zur tatsächlichen Startadresse eingeben.

Beispielsweise steht ein Programm im Speicher ab Adresse 6000 hex, soll aber so gespeichert werden, daß der Basic-Lader es auf 4000 hex erzeugt. Die Eingaben sind für diesen Fall:

`Startadresse=&4000, Offset=&2000.`

Wenn Sie kein Offset benötigen, drücken Sie einfach <ENTER>. Jetzt fehlt nur noch die Nummer der ersten Zeile und die Schrittweite der Numerierung, bevor »CPC« mit der Erzeugung des Laders beginnt und ihn als ASCII-Datei auf Kassette oder Diskette speichert.

Das Menü »Parameter bestimmen« bricht man mit <ESC> oder <CTRL+C> ab und beendet es mit <ENTER>. Die Tasten 1 bis 8 ändern einzelne Parameter:
<1> Zeilenadresse dezimal oder hexadezimal anzeigen.
<2> Byte dezimal oder hexadezimal erwarten und anzeigen.
<3> Anzahl der Byte pro Zeile. Eingabe in Form einer Zahl kleiner oder gleich 128.
<4> Ende annehmen ist normalerweise »Ja«. Haben Sie »zwei Ziffern« bei hexadezimaler oder »drei Ziffern« bei dezimaler Eingabe gewählt, geht »CPC« automatisch zum nächsten Byte über. Ist dieser Punkt »Nein«, beenden Sie jedes Byte mit der Leertaste, <ENTER> oder dem Punkt.
<5> Prüfsumme abfragen. »Nein« ist nur bei Basic-Ladern ohne Prüfsumme zu empfehlen.
<6> Prüfsumme dezimal oder hexadezimal erwarten und anzeigen.
<7> Länge der Prüfsumme. Minimal 4 bei hexadezimaler und 5 bei dezimaler Ausgabe (Voreinstellung). »CPC« benötigt diese Angabe für das Erzeugen der Basic-Lader.
<8> Prüfsummentyp. Viele Basic-Lader verwenden Prüfsummen des Typs »Add«, also eine einfache Addition aller Byte einer Zeile. Die »CPC«-Lader verfügen jedoch über eine Hash-Prüfsumme, die Fehler und Vertauschungen erkennt. Falscheingaben sind hier fast unmöglich.

Während der Arbeit mit »CPC« beenden Sie jede Eingabe mit <ENTER> und korrigieren mit <DEL>. Für Dateinamen müssen Diskettenbenutzer ein gültiges Format wählen. Bei allen Zahleneingaben ist eine dezimale oder hexadezimale Eingabe mit vorangestelltem »$«, »#« oder »&« wählbar. An jeder Stelle, an der eine Taste zur Bestätigung zu drücken ist, läßt sich die Funktion durch <ESC> oder <CTRL+C> abbrechen. (Stefan Aust/ja)

Steckbrief	
Programm:	CPC
Computer:	CPC 464/664/6128
Checksummer:	Explora
Datenträger:	Kassette/Diskette

```
1000 '                                        [9110]
1010 '--> "CPC = Comfortable Program for
      Codeinput" by St. M. Aust              [53B0]
1020 '                                        [8F14]
1030 '                                        [9016]
1040 '--> Initialisierung                     [CFE8]
1050 '                                        [B61A]
1060 CLOSEIN:MEMORY 19999                     [AC78]
1070 KEY DEF 66,0,3:CALL &BB48                [9D26]
1080 brk$=CHR$(3):bell$=CHR$(7):back$=CH
     R$(8):cr$=CHR$(13):clr$=CHR$(16)         [D184]
1090 del$=back$+clr$:cur1$=CHR$(143)+bac
     k$:cur2$=CHR$(211)+back$                 [A504]
1100 dz$="0123456789":hx$="0123456789ABC
     DEF"                                     [0BA6]
1110 DIM b$(128),fl(8):fl(1)=1:fl(2)=1:f
     l(3)=8:fl(5)=1:fl(6)=1:fl(7)=4:fl(8
     )=1                                      [52BA]
1120 ENV 1,15,-1,20:ENV 2,15,-1,4             [2772]
1130 '                                        [9018]
1140 '--> Menue                               [BA7E]
1150 '                                        [921C]
1160 MODE 1:PAPER 0:PEN 1:PAPER#1,0:PEN#
     1,1:PAPER#2,0:PEN#2,3                    [3464]
1170 LOCATE 18,1:PRINT"C P C"                 [7450]
1180 LOCATE 4,2:PRINT"Comfortable Progra
     m for Codeinput"                         [01EE]
1190 LOCATE 6,7:PEN 1:PRINT"Geschrieben
     von Stefan M.Aust"                       [B9C0]
1200 LOCATE 8,8:PRINT"Version II - Oktob
     er 1986"                                 [7E6A]
1210 SOUND 1,400,0,15,1:SOUND 2,450,0,15
     ,1:SOUND 4,500,0,0,1                     [C3B6]
1220 WHILE SQ(1)<>4:WEND                      [B074]
1230 x=6:y=6:xl=29:yl=15:GOSUB 3310:WIND
     OW SWAP 1                                [0EBC]
1240 LOCATE 9,3:PRINT"(L)ade Code"            [A788]
1250 LOCATE 7,5:PRINT"(S)chreibe Code"        [F2E8]
1260 LOCATE 7,7:PRINT"(C)ode eingeben"        [E11E]
1270 LOCATE 7,9:PRINT"(E)rzeuge DATAs"        [267C]
1280 LOCATE 9,11:PRINT"(B)eende CPC"          [B93E]
1290 WINDOW#2,1,40,23,23:PRINT#2,TAB(12)
     "Bitte waehlen: ";cur1$;                 [F4D2]
1300 GOSUB 3410:p=INSTR("LSCEB"+brk$,a$)
     :IF p=0 THEN PRINT bell$;:GOTO 1300
                                              [566E]
1310 x=3:y=17:xl=18:yl=3:GOSUB 3310           [2432]
1320 PRINT#2,a$:ON p GOTO 1330,1420,1520
     ,1880,2670,2670                          [7CF4]
1330 '                                        [901C]
1340 '--> Lade Code                           [4BB0]
1350 '                                        [B620]
1360 PRINT#1,"<2>(L)ade Code"                 [EDBA]
1370 WINDOW SWAP 2:PRINT TAB(10)"Name: "
     ;:GOSUB 2860:n$=b$                       [5CE6]
1380 PRINT TAB(11)"Startadresse: ";:GOSU
     B 2750:start=b                           [8240]
1390 PRINT"Disk/Kassette einlegen & Tast
     e druecken"                              [7CF6]
1400 GOSUB 3410:IF brk THEN 1130              [FC52]
1410 LOAD"!"+n$,start:GOTO 1130               [4EB4]
1420 '                                        [971C]
1430 '--> Schreibe Code                       [7F0E]
1440 '                                        [9520]
1450 PRINT#1,"(S)chreibe Code"                [3798]
1460 WINDOW SWAP 2:PRINT TAB(10)"Name: "
     ;:GOSUB 2860:n$=b$                       [13E6]
1470 PRINT TAB(11)"Startadresse: ";:GOSU
     B 2750:start=b                           [0F40]
1480 PRINT TAB(12)"Endadresse: ";:GOSUB
     2750:finish=b                            [D5FC]
1490 PRINT"Disk/Kassette einlegen & Tast
     e druecken"                              [01F8]
1500 GOSUB 3410:IF brk THEN 1130              [DA54]
1510 SAVE"!"+n$,b,start,finish-start:GOT
     O 1130                                   [0D00]
1520 '                                        [B91E]
1530 '--> Code eingeben                       [CB40]
1540 '                                        [9322]
1550 PRINT#1,"(C)ode einqeben"                [48CA]
1560 GOSUB 2470:IF brk THEN 1130 ELSE WI
     NDOW SWAP 2                              [1A46]
1570 PRINT TAB(11)"Startnummer: ";:GOSUB
      2750:nr=b                               [BC00]
1580 PRINT TAB(10)"Schrittweite: ";:GOSU
     B 2750:inc=b                             [B96E]
1590 PRINT TAB(10)"Startadresse: ";:GOSU
     B 2750:start=b                           [6044]
1600 GOSUB 3230:LOCATE#1,32,2:PRINT#1,"(
     C)ode eingeben"                          [D770]
1610 IF fl(2)THEN h$="&":l=2 ELSE h$="":
     l=3                                      [7F1E]
1620 h=l+1:xa=16:xe=xa+h*15                   [37BA]
1630 PRINT:PRINT USING"(#####) ";nr;:IF
     fl(1)THEN PRINT" "HEX$(start,4)": "
     ;ELSE PRINT USING"#####: ";start;        [A692]
1640 FOR i=0 TO fl(3):b$(i)="":NEXT           [BAEA]
1650 p=0:x=xa                                 [1A22]
1660 y=VPOS(#0):IF x<xa THEN x=xe:y=y-1
     ELSE IF x>xe THEN PRINT CHR$(10)CHR
     $(11):x=xa:y=VPOS(#0)                    [DD3E]
1670 b$=b$(p):LOCATE x,y:PRINT b$;            [5B9E]
1680 lmax=l:IF fl(2)THEN GOSUB 2950 ELSE
      GOSUB 2970                              [0644]
1690 IF fl(2)=0 THEN b=VAL(h$+b$):IF b>2
     55 THEN PRINT bell$;:GOTO 1680           [7202]
1700 IF brk THEN 1130 ELSE IF del=0 THEN
      1730                                    [AEFC]
1710 b$(p)="":IF p>0 THEN p=p-1:x=x-h EL
     SE del=0:PRINT bell$;                    [049C]
1720 GOTO 1660                                [0A20]
1730 LOCATE x,y:PRINT b$;:b$(p)=b$            [0698]
1740 x=x+h:p=p+1:IF p<fl(3)THEN 1660          [6ECC]
1750 FOR i=0 TO fl(3)-1:POKE start+i,VAL
     (h$+b$(i)):NEXT                          [24F8]
1760 IF fl(5)=0 THEN 1830                     [EA3A]
1770 GOSUB 3480:LOCATE x,y:PRINT"= ";:x=
     x+2:b$="":lmax=fl(7)                     [A670]
1780 IF fl(6)THEN GOSUB 2950 ELSE GOSUB
     2970                                     [5124]
1790 IF brk THEN 1130 ELSE IF del THEN x
     =x-2:PRINT del$;del$;:GOTO 1710          [AD8C]
1800 LOCATE x,y:PRINT b$;                     [A10C]
1810 IF fl(6)THEN pr2=VAL("&"+b$)ELSE pr
     2=VAL(b$)                                [766A]
1820 IF pr<>pr2 THEN PRINT bell$;:GOTO 1
     780                                      [CE0E]
1830 start=start+fl(3):nr=nr+inc              [2C44]
1840 SOUND 1,400,0,15,2                       [B1A6]
1850 GOTO 1630                                [2422]
1860 b$(p)="":IF p>1 THEN p=p-1:x=x-h EL
     SE del=0:PRINT bell$;                    [A0AA]
1870 GOTO 1660                                [752C]
1880 '                                        [9930]
1890 '--> Erzeuge DATA's                      [7CF8]
1900 '                                        [BF22]
1910 PRINT#1,"(E)rzeuge DATAs"                [B322]
1920 GOSUB 2470:IF brk THEN 1130              [BC6A]
1930 x=2:y=7:xl=37:yl=9:GOSUB 3310:WINDO
     W SWAP 1                                 [A468]
1940 LOCATE 2,1:PRINT"DATA-File-Name: ";
     :GOSUB 2860:dn$=b$                       [680E]
1950 LOCATE 2,2:PRINT"Code-File-Name: ";
     :GOSUB 2860:cn$=b$                       [D2D2]
1960 LOCATE 4,4:PRINT"Startadresse: ";:G
     OSUB 2750:start=b                        [BC40]
1970 LOCATE 6,5:PRINT"Endadresse: ";:GOS
     UB 2750:finish=b                         [CC00]
1980 LOCATE 10,7:PRINT"Offset: ";:GOSUB
     2750:offs=b                              [FE88]
1990 x=2:y=17:xl=37:yl=4:GOSUB 3310:WIND
     OW SWAP 1                                [86CC]
2000 LOCATE 4,1:PRINT"Erste Nummer: ";:G
     OSUB 2750:nr=b                           [7DC6]
2010 LOCATE 4,2:PRINT"Schrittweite: ";:G
     OSUB 2750:inc=b                          [B34E]
2020 PRINT#2,"Disk/Kassette einlegen & T
     aste druecken"                           [48E6]
2030 GOSUB 3410:IF brk THEN 1130              [FE52]
2040 form=1:IF form THEN nn$="00000"ELSE
      nn$=""                                  [B09E]
2050 st2=start:start=start+offs:finish=f
     inish+offs:nr2=nr+4*inc                  [712A]
2060 d1=6:IF fl(5)THEN d1=d1+5+(fl(8)=0)
                                              [EC16]
2070 d2=5:IF fl(5)THEN d2=d2+5+(fl(8)=0)
                                              [341C]
2080 PRINT#2,"Erzeugung beginnt...";          [8888]
2090 OPENOUT"!"+dn$                           [E436]
2100 h$="* "+dn$+" - DATA-Lader von 'CPC
     ' *"                                     [13CE]
2110 PRINT#9,nr;"'"+STRING$(LEN(h$),42):
     nr=nr+inc                                [A4C0]
2120 PRINT#9,nr;"'"+h$:nr=nr+inc              [92A6]
2130 PRINT#9,nr;"'"+STRING$(LEN(h$),42):
     nr=nr+inc                                [48C4]
2140 PRINT#9,nr;"'":nr=nr+inc                 [473C]
2150 PRINT#9,nr;"DATA ";:nr=nr+inc            [37DA]
2160 IF fl(1)THEN PRINT#9,HEX$(start-off
     s,4);ELSE PRINT#9,USING"#####";star
     t-offs;                                  [8548]
2170 FOR i=0 TO fl(3)-1:IF fl(2)THEN IF
     form THEN PRINT#9,",";HEX$(PEEK(sta
     rt+i),2);:GOTO 2190 ELSE PRINT#9,",
     ";HEX$(PEEK(start+i));:GOTO 2190         [33B8]
2180 a$=STR$(PEEK(start+i)):PRINT#9,",";
     RIGHT$(nn$+MID$(a$,2),3);                [206E]
2190 NEXT:IF fl(5)=0 THEN PRINT#9:GOTO 2
     220                                      [9514]
2200 GOSUB 3480:IF fl(6)THEN PRINT#9,","
     ;HEX$(pr,fl(7)):GOTO 2220                [8F7E]
2210 a$=STR$(pr):PRINT#9,",";RIGHT$(nn$+
     MID$(a$,2),fl(7))                        [958E]
2220 IF INKEY$=brk$THEN CALL &BC92:GOTO
     1130                                     [94A4]
2230 start=start+fl(3):IF start<finish T
     HEN 2150                                 [7318]
2240 PRINT#9,nr;"DATA *ENDE*":nr=nr+inc       [4C44]
2250 PRINT#9,nr;"adr=";:IF fl(1)THEN PRI
     NT#9,"&"+HEX$(st2,4);ELSE PRINT#9,M
```

Listing 2. Eingabe von Maschinencode im Eiltempo

```
     ID$(STR$(st2),2);                      [513A]
2260 PRINT#9,":zeile="+MID$(STR$(nr2),2)
     ;:nr=nr+inc                            [CB04]
2270 PRINT#9,":MEMORY ";:IF st2=32768 TH
     EN PRINT#9,"&7FFF"ELSE PRINT#9,"adr
     -1"                                    [9752]
2280 PRINT#9,nr;"READ d$:IF d$="+CHR$(34
     )+"*ENDE*"+CHR$(34)+"THEN";nr+d1*in
     c:nr=nr+inc                            [61E4]
2290 IF fl(5)THEN PRINT#9,nr;"pr=0":nr=n
     r+inc                                  [2544]
2300 PRINT#9,nr;"FOR i=1TO";fl(3):nr=nr+
     inc                                    [E40E]
2310 IF fl(2)THEN PRINT#9,nr;"READ a$:a=
     VAL("+CHR$(34)+"&"+CHR$(34)+"+a$)":
     nr=nr+inc ELSE PRINT#9,nr;"READ a":
     nr=nr+inc                              [E450]
2320 PRINT#9,nr;"POKE adr,a:adr=adr+1":n
     r=nr+inc                               [C496]
2330 IF fl(5)=0 THEN 2370 ELSE IF fl(8)T
     HEN 2350                               [A146]
2340 PRINT#9,nr;"pr=pr+a":nr=nr+inc:GOTO
      2370                                  [BACA]
2350 PRINT#9,nr;"pr=pr*2:IF pr>65535THEN
      pr=pr-65535":nr=nr+inc                [C5DA]
2360 PRINT#9,nr;"pr=UNT(pr)XOR a:if pr<0
     THEN pr=pr+65536":nr=nr+inc            [6B72]
2370 PRINT#9,nr;"NEXT i":nr=nr+inc          [3088]
2380 IF fl(5)=0 THEN 2420                   [7830]
2390 IF fl(6)THEN PRINT#9,nr;"READ pr$:p
     r2=VAL("+CHR$(34)+"&"+CHR$(34)+"+pr
     $):if pr2<0then pr2=pr2+65536":nr=n
     r+inc:GOTO 2410                        [E97C]
2400 PRINT#9,nr;"READ pr2":nr=nr+inc        [1A8C]
2410 PRINT#9,nr;"IF pr<>pr2 THEN PRINT"+
     CHR$(34)+"Pruefsummenfehler in Zeil
     e"+CHR$(34)+";zeile:STOP":nr=nr+inc
                                            [9F52]
2420 PRINT#9,nr;"zeile=zeile+";MID$(STR$
     (inc),2);":GOTO";nr-d2*inc:nr=nr+in
     c                                      [6BAC]
2430 PRINT#9,nr;"SAVE"+CHR$(34)+cn$+CHR$
     (34)+",B,&"+HEX$(st2)+",&"+HEX$(fin
     ish-st2):nr=nr+inc                     [C3D4]
2440 PRINT#9,nr;"PRINT d$:END":CLOSEOUT     [BC98]
2450 PRINT#2," und ist fertig.";bell$       [70C2]
2460 GOSUB 3410:GOTO 1130                   [4658]
2470 '                                      [B728]
2480 '--> Parameter einstellen              [3B82]
2490 '                                      [B92C]
2500 x=10:y=8:xl=28:yl=14:GOSUB 3310        [5098]
2510 WINDOW SWAP 0,1:PRINT"Parameter bes
     timmen:"                               [2A24]
2520 LOCATE 1,3:SOUND 1,900,0,15,2          [DCEC]
2530 PRINT"(1) Adresse --------> ";:IF fl
     (1)THEN PRINT"hex"ELSE PRINT"dez"      [AA2E]
2540 PRINT"(2) Bytes ----------> ";:IF fl
     (2)THEN PRINT"hex"ELSE PRINT"dez"      [6968]
2550 PRINT"(3) Byteanzahl ---->";fl(3)      [4B2C]
2560 PRINT"(4) Ende annehmen -> ";:IF fl
     (4)THEN PRINT"nein"ELSE PRINT"ja<2>
     "                                      [B2BC]
2570 PRINT"(5) Pruefsumme ----> ";:IF fl
     (5)THEN PRINT"ja<2>"ELSE PRINT"nein
     "                                      [3456]
2580 PRINT"(6) Pruefsumme ----> ";:IF fl
     (6)THEN PRINT"hex"ELSE PRINT"dez"      [7B02]
2590 PRINT"(7) deren Laenge -->";fl(7)      [6D60]
2600 PRINT"(8) und Typ --------> ";:IF fl
     (8)THEN PRINT"hash"ELSE PRINT"add "
                                            [BDCA]
2610 PRINT:PRINT"Mit <ENTER> geht's weit
     er"                                    [FAAA]
2620 PRINT#2,TAB(11)"(1-8) oder ENTER:_"
     ;cur1$;                                [9A86]
2630 GOSUB 3410:IF a$=brk$OR a$=cr$ THEN
      RETURN                                [7E58]
2640 IF a$<"1"OR a$>"8"THEN PRINT bell$;
     :GOTO 2630 ELSE PRINT#2,a$:f=VAL(a$
     )                                      [5276]
2650 IF f<>3 AND f<>7 THEN fl(f)=1-fl(f)
     :GOTO 2520                             [49A2]
2660 WINDOW SWAP 0,2:PRINT TAB(12)"neuer
      Wert: ";:GOSUB 2750:fl(f)=b:WINDOW
      SWAP 0,2:GOTO 2520                    [6DF8]
2670 '                                      [972C]
2680 '--> Beende CPC                        [FD10]
2690 '                                      [BD30]
2700 PRINT#1,"<2>(B)eende CPC"              [D808]
2710 WINDOW SWAP 2:PRINT TAB(9)"Zurueck
     mit ESC oder ^C"                       [01A8]
2720 GOSUB 3410:IF brk THEN 1130            [FD5E]
2730 MODE 2                                 [3EC6]
2740 END                                    [F888]
2750 '                                      [952A]
2760 '--> Sub: Hex-Dez-Input                [E208]
2770 '                                      [972E]
2780 b$="":b=0:lmax=5                       [C5BE]
2790 GOSUB 3150:IF a=13 AND b$<>"&"THEN
     2840                                   [60C0]
2800 IF a=127 THEN GOSUB 3200:GOTO 2790     [C814]
2810 IF b=0 THEN IF a>34 AND a<39 THEN i
     n$=hx$:b$="&":b=1:PRINT a$;:GOTO 27
     90 ELSE in$=dz$                        [2EE8]
2820 IF INSTR(in$,a$)>0 AND b<lmax THEN
     b$=b$+a$:b=b+1:PRINT a$;ELSE PRINT
     bell$;                                 [5426]
2830 GOTO 2790                              [5630]
2840 b=VAL(b$):IF b<0 THEN b=b+65536        [9D44]
2850 RETURN                                 [839E]
2860 '                                      [C82E]
2870 '--> Sub: Filename-Input               [3C64]
2880 '                                      [C232]
2890 b$="":b=0:lmax=16                      [E926]
2900 GOSUB 3150:IF a=13 THEN RETURN         [C41C]
2910 IF a=127 THEN GOSUB 3200:GOTO 2900     [700A]
2920 IF a>31 AND a<127 AND b<lmax THEN b
     $=b$+a$:b=b+1:PRINT a$;ELSE PRINT b
     ell$;                                  [0BCC]
2930 GOTO 2900                              [6324]
2940 '                                      [C82C]
2950 '--> Sub: Hex-Zahl eingeben            [8122]
2960 in$=hx$:GOTO 2990                      [6128]
2970 '--> Sub: Dez-Zahl eingeben            [1F22]
2980 in$=dz$:GOTO 2990                      [1128]
2990 '                                      [C536]
3000 '--> Sub: Input                        [45AC]
3010 '                                      [B816]
3020 b=LEN(b$):brk=0:IF del THEN del=0:G
     OTO 3100 ELSE del=0                    [BF62]
3030 GOSUB 3150                             [A69E]
3040 IF a=3 THEN brk=1:GOTO 3120            [46B8]
3050 IF a=13 OR a=32 OR a=46 THEN 3120      [9936]
3060 IF a=47 OR a=92 THEN b$=bb$:b=LEN(b
     b$):RETURN                             [6960]
3070 IF a=127 THEN 3100                     [4708]
3080 IF INSTR(in$,a$)>0 AND b<lmax THEN
     b$=b$+a$:b=b+1:PRINT a$;ELSE PRINT
     bell$;                                 [0624]
3090 IF fl(4)THEN 3030 ELSE IF b=lmax TH
     EN 3120 ELSE 3030                      [59A8]
3100 IF b>0 THEN GOSUB 3200:GOTO 3030       [E120]
3110 PRINT bell$;:del=1                     [B2DA]
3120 b$=RIGHT$("00000"+b$,lmax)             [74BE]
3130 bb$=b$:RETURN                          [A058]
3140 '                                      [B91E]
3150 '--> Sub: Tastdruck nach a,a$          [80DA]
3160 PRINT cur2$;                           [D0E4]
3170 a$=UPPER$(INKEY$):IF a$=""THEN 3170
                                            [2FB6]
3180 PRINT clr$;                            [E872]
3190 a=ASC(a$):RETURN                       [7BA4]
3200 '--> Sub: DEL-Routine                  [B460]
3210 IF b>0 THEN b=b-1:b$=LEFT$(b$,b):PR
     INT back$;clr$;ELSE PRINT bell$;       [359E]
3220 RETURN                                 [878E]
3230 '                                      [921E]
3240 '--> Sub: Rahmen zeichnen              [1936]
3250 '                                      [B822]
3260 MODE 2                                 [2BC4]
3270 MOVE 112,352:DRAW 527,352:DRAW 527,
     399:DRAW 112,399:DRAW 112,352          [BD9E]
3280 MOVE 116,356:DRAW 523,356:DRAW 523,
     395:DRAW 116,395:DRAW 116,356          [81B0]
3290 LOCATE 28,25:PRINT"Zurueck mit ESC
     oder ^C"                               [F34E]
3300 WINDOW 1,80,4,24:RETURN                [325E]
3310 '                                      [901C]
3320 '--> Sub: Fenster oeffnen              [8046]
3330 '                                      [9220]
3340 WINDOW#1,x,x+xl-1,y,y+yl-1:CLS#1       [51E6]
3350 xp=x*16-16:yp=415-y*16:xm=xl*16-1:y
     m=yl*16-1                              [397C]
3360 PLOT xp,yp,1:DRAWR xm,0:DRAWR 0,-ym
     :DRAWR -xm,0:DRAWR 0,ym                [5FDA]
3370 xp=xp+4:yp=yp-4:xm=xm-8:ym=ym-8        [85A2]
3380 PLOT xp,yp,3:DRAWR xm,0:DRAWR 0,-ym
     :DRAWR -xm,0:DRAWR 0,ym                [27E2]
3390 WINDOW#1,x+1,x+xl-2,y+1,y+yl-2         [5984]
3400 RETURN                                 [7D8E]
3410 '                                      [C01E]
3420 '--> Sub: Auf Taste warten             [F074]
3430 '                                      [C222]
3440 WHILE INKEY$<>"":WEND                  [A79C]
3450 a$=UPPER$(INKEY$):IF a$=""THEN 3450
                                            [87BA]
3460 IF a$=brk$THEN brk=1 ELSE brk=0        [50A4]
3470 RETURN                                 [9F9C]
3480 '                                      [972C]
3490 '--> Sub: Checksum bilden              [0428]
3500 '                                      [BD1E]
3510 pr=0:IF fl(8)THEN 3540                 [8D2E]
3520 FOR i=0 TO fl(3)-1:pr=pr+PEEK(start
     +i):NEXT                               [49EA]
3530 RETURN                                 [8C96]
3540 FOR i=0 TO fl(3)-1:pr=pr*2:IF pr>65
     535 THEN pr=pr-65535                   [BA5A]
3550 pr=UNT(pr)XOR PEEK(start+i):IF pr<0
      THEN pr=pr+65536                      [F3C4]
3560 NEXT:RETURN                            [2B8E]
```

Listing 2. (Schluß)

Der CPC hat Töne

Auch musikalisch haben die Computer der Schneider CPC-Serie einiges auf dem Kasten. Nur fällt es meist schwer, ihnen diese Fähigkeiten zu entlocken. Der »Envelope Composer« nimmt Ihnen viel Fleißarbeit ab.

An Basic-Befehlen - auch zur Tonerzeugung beziehungsweise -beeinflussung - ist der CPC eigentlich reich gesegnet. Will man aber beispielsweise eigene Lautstärke-Hüllkurven entwerfen, läßt einen der Computer im Stich: Der Griff zu Bleistift und Papier ist dann unausweichlich. Vielmehr war er es, denn das Programm »Envelope Composer« erleichtert nun diese Aufgabe. Zukünftig entwickeln Sie damit die Hüllkurven am Bildschirm. Dazu stellt sie der Computer in einem Gitterraster grafisch dar. Der sofortigen Erfolgskontrolle dient das probeweise Abspielen der Kreation. Ist die endgültige Form gefunden, speichert der CPC die Kurven wahlweise als Basic-Programm in Form von ENV-Befehlen. Auch an eine Druckerausgabe ist gedacht. Da sich eine Kurve über mehrere Bildschirmseiten erstrecken kann, sind Hardcopies sowohl von einzeln bearbeiteten Teilen (Bild 1) als auch vom gesamten Kurvenverlauf (Bild 2) anzufertigen.

Geben Sie zunächst Listing 1 ein und speichern es. Auch Listing 2 speichern Sie sicherheitshalber gleich nach der Eingabe. Danach starten Sie es mit »RUN«. Es erzeugt dann selbsttätig den Maschinencode für die benötigte Befehlserweiterung und speichert ihn automatisch unter dem Namen »ENVELOPE.BIN«. Mit Listing 3 verfahren Sie genauso; es enthält die Hardcopyroutine, die den Dateinamen »ENVELOPE.HRD« erhält. Dabei handelt es sich um eine Routine die wir erstmals in der Happy-Computer, Ausgabe 6/86, veröffentlichten. Sie ist in dieser Version an den DMP 2000 und andere Epson-kompatible Drucker angepaßt. Die Anpassung an andere Drucker entnehmen Sie

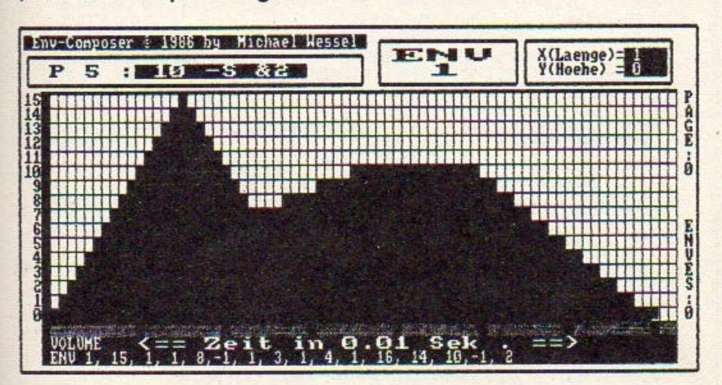


Bild 1. So einfach und übersichtlich können auch Sie in Zukunft Ihre Lautstärke-Hüllkurven entwickeln

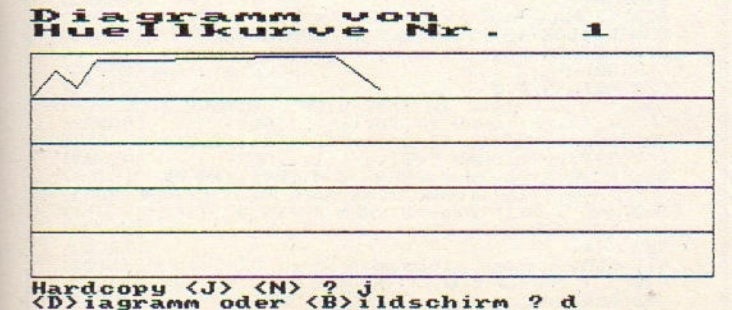


Bild 2. Der gesamte Kurvenzug wird im Diagramm sichtbar

Bild 3. Aus dem Hauptmenü heraus ist nicht nur die hörbare Kontrolle Ihrer Kreation wählbar

bei Bedarf bitte diesem Beitrag. Natürlich lassen sich auch andere Hardcopyroutinen einbinden. Voraussetzung dafür ist jedoch, daß sie oberhalb der Adresse 9FFF hex liegen. Die genaue Adresse des Aufrufs müssen Sie dann im vierten und fünften Byte der Zeile 109 in Listing 2 eintragen (dort steht in dieser Version A01B hex in der Reihenfolge Lowbyte, Highbyte).

Dem Start des Programms folgt eine kurze Verzögerung, während der die beiden Binärdateien nachladen. Am besten eignet sich ein Eingabebeispiel zum Kennenlernen: Oben links erscheint der Schriftzug

`Env Nr. :`

und der Cursor dahinter signalisiert, daß der CPC auf Ihre Eingabe wartet. Nun wählen Sie eine der möglichen 15 Hüllkurven mit ihrer Nummer aus. Geben Sie also zur Probe <1> ein und drücken Sie danach <ENTER>. Die nächste Aufforderung

`Start-Lautstärke der Envelope setzen`

wird nun unten links sichtbar. Die Lautstärken sind am linken Bildschirmrand aufsteigend abzulesen. Mit den Richtungstasten bewegen Sie den Cursor auf die gewünschte Lautstärke. Fahren Sie also den Cursor auf die Position 0 (Anzeige »Y=0«, oben links im Bild 1) und drücken Sie an dieser Stelle <COPY> zur Bestätigung. Nun ist der Cursor frei beweglich. Um in unserem Beispiel fortzufahren, bringen Sie den Cursor ans obere Ende der Skala (auf den Wert 15). Anschließend drücken Sie die rechte Cursorsteuertaste, worauf der Zeiger gleich um einige Positionen nach rechts springt. Nur auf diesen Positionen ist die Lage des neuen Kurvenpunkts erlaubt. Das signalisieren auch stets die beiden Anzeigen »X(Laenge)=« und »Y(Hoehe)=« oben rechts auf dem Monitor, wenn sie invers (weiße Zahl auf schwarzem Hintergrund) dargestellt sind. Nachdem Sie nun an der ersten möglichen Position wieder <COPY> drücken, zeichnet Ihr CPC den ersten Teil der Grafik. Daraufhin ändern sich noch weitere Bildschirmanzeigen. Oben links sehen Sie nun nämlich die Werte dieses ersten Hüllkurventeils (Part). In der untersten Zeile ist der entsprechende ENV-Befehl mit den bisherigen Parametern sichtbar. Nun bewegen Sie den Cursor mit gleichbleibendem x-Wert auf die y-Ordinate 0 (ganz unten). Wenn Sie nun einmal die rechte Cursortaste drücken und im Anschluß daran <COPY>, erhalten Sie eine Hüllkurve in Form eines Dreiecks. Sollen sich spätere Kurven über mehrere Bildschirme hinweg erstrecken, gelangen Sie durch Druck der Leertaste in diese weiteren Bereiche.

Die Kurve ist vollständig und deshalb nicht mehr zu erweitern, wenn die Parameter des ENV-Befehls am unteren Bildschirmrand vollzählig sind. Drücken Sie jetzt

<ENTER>, um ins Hauptmenü zu gelangen (Bild 3). Der Cursor steht auf dem Punkt »Spielen«. In diesem Menü löst die ENTER-Taste die Funktionen aus. Probieren Sie es also einfach mal mit <ENTER>, und Sie hören mit dem Kammerton A, was Sie gerade eingegeben haben. Die anderen Unterpunkte erreichen Sie, indem Sie den Cursor mit der oberen und unteren Cursor-Steuertaste bewegen. Beim Sichern erzeugt der Envelope Composer ein Basic-Programm mit den generierten ENV-Befehlen. Diese dienen später als Grundstock für eigene Musikprogramme.

»Helppage« ruft ein Hilfsmenü auf (Bild 4), das die wichtigsten Bedienungsregeln in Erinnerung ruft. Mit »Diagramm« erhalten Sie nicht nur einen Überblick über die gesamte Hüllkurve (wie in Bild 2); hier läßt sich auch die Hardcopy aufrufen. Wollen Sie eine weitere Hüllkurve entwickeln, bedienen Sie sich des Menüpunkts »Ende«. Das Ende der Arbeit leiten Sie über »Programm verlassen« oder durch zweimaligen Druck der Taste <ESC> ein.

Das Programm nutzt – wie übrigens auch die vier Abbildungen zeigen – die verschiedenen Schriftgrößen aller drei Bildschirmmodi gleichzeitig. Da es auf einem CPC 464 entwickelt wurde, arbeitet diese Ausgabeart nicht auf den CPCs 664 und 6128. Die Zeichen erscheinen auf Computern dieser Bauserie in gewohnter Modus-2-Darstellung. Andere Einschränkungen existieren jedoch nicht.

(Michael Wessel/ja)

Bild 4. Haben Sie im Eifer des Gefechts einmal die Bedienung vergessen, hilft Ihnen der Aufruf der »Help-Page« auf die Sprünge

Steckbrief	
Programm:	Envelope Composer
Computer:	CPC 464, mit Einschränkungen auch auf 664/6128
Checksummer:	Explora/CPC
Datenträger:	Kassette, Diskette

```
10 '****************************           [3080]
20 '*    Envelope - Composer    *           [9CA0]
30 '*  Michael Wessel , 1986    *           [AB50]
40 '*  2000  Hamburg Niendorf   *           [9F38]
50 '****************************           [5488]
60 '                                        [845A]
70 'Arrays definieren                       [86B2]
80 '                                        [865E]
90 ON BREAK GOSUB 3530                      [4B6C]
100 MEMORY &4FFF:DEFINT a-z:DEFREAL s       [F464]
110 DIM t(4,14),s(4,14),l(4,14)             [6D32]
120 DIM erlau(14),le(14)                    [035E]
130 LOAD"envelope.bin                       [0876]
140 LOAD"envelope.hrd                       [E282]
150 CALL &5001                              [5ABC]
160 svscr=&5116                             [8E30]
170 '                                       [03BE]
180 'Variablen definieren                   [A15A]
190 '                                       [05C2]
200 part=-1                                 [2008]
210 y=21:x=0                                [F6D6]
220 csteig=1                                [6142]
230 page=0                                  [B77E]
240 ofs=0                                   [CAD6]
250 ax=0:ay=0                               [6AFC]
260 '                                       [E2BE]
270 'Screenparameter                        [BA02]
280 '                                       [0CC2]
290 MODE 2                                  [2764]
300 INK 0,13                                [B1EA]
310 BORDER 13                               [33EC]
320 INK 1,13                                [80F0]
330 PEN 1                                   [34D4]
340 '                                       [06BC]
350 'Windows definieren                     [DC06]
360 '                                       [04C0]
370 WINDOW #1,72,76,3,3:WINDOW #2,72,76,
    4,4                                     [45CE]
380 WINDOW #5,3,20,4,4: WINDOW #6,3,38,4
    ,4: WINDOW #7,4,77,24,24                [6FC6]
390 PAPER #7,1:PEN #7,0                     [07FC]
400 PAPER #4,1:PEN #4,0                     [D9E0]
410 PAPER #2,1:PEN #2,0                     [FADA]
420 '                                       [E6BA]
430 'Rahmen zeichen                         [657E]
440 '                                       [E4BE]
450 PRINT CHR$(150)+STRING$(78,154)+CHR$
    (156)                                   [A526]
460 FOR iy=2 TO 24                          [FDB6]
470 LOCATE 1,iy:PRINT CHR$(149):LOCATE 8
    0,iy:PRINT CHR$(149)                    [5210]
480 NEXT                                    [AEF6]
490 LOCATE 1,25:PRINT CHR$(147)+STRING$(
    78,154)+CHR$(153);                      [E056]
500 LOCATE 60,2:PRINT CHR$(150)+STRING$(
    16,154)+CHR$(156):LOCATE 60,3:PRINT
    CHR$(149)+STRING$(16,32)+CHR$(149):L
    OCATE 60,4:PRINT CHR$(149)+STRING$(1
    6,32)+CHR$(149):LOCATE 60,5:PRINT CH
    R$(147)+STRING$(16,154)+CHR$(153)       [256C]
510 LOCATE 43,2:PRINT CHR$(150)+STRING$(
    15,154)+CHR$(156):LOCATE 43,3:PRINT
    CHR$(149)+STRING$(15,32)+CHR$(149):L
    OCATE 43,4:PRINT CHR$(149)+STRING$(1
    5,32)+CHR$(149):LOCATE 43,5:PRINT CH
    R$(147)+STRING$(15,154)+CHR$(153)       [206E]
520 LOCATE 2,3:PRINT CHR$(150)+STRING$(3
    8,154)+CHR$(156):LOCATE 2,4:PRINT CH
    R$(149)+STRING$(38,32)+CHR$(149):LOC
    ATE 2,5:PRINT CHR$(147)+STRING$(38,1
    54)+CHR$(153)                           [53E2]
530 '                                       [07BE]
540 'x-Leiste der Grafik                    [5F74]
550 '                                       [E1C2]
560 |XON                                    [0158]
570 FOR ix=4 TO 77                          [51CC]
580 LOCATE ix,22:PRINT CHR$(207)            [91B0]
590 IF LEN(STR$(ix/10))<3 THEN LOCATE ix
    +5,22:PRINT CHR$(143)                   [06A8]
600 NEXT                                    [7CEA]
610 |XOFF                                   [C6CC]
620 '                                       [0EBE]
630 'y-Leiste der Grafik                    [A476]
640 '                                       [10C2]
650 FOR iy=21 TO 6 STEP-1                   [202E]
660 LOCATE 2,iy:PRINT USING"##";16-(iy-5
    );:PRINT CHR$(143)                      [DE76]
670 NEXT                                    [1CF8]
680 LOCATE 4,22:PRINT CHR$(143):LOCATE 4
    ,21:PRINT CHR$(143)                     [99A2]
690 '                                       [E5CC]
700 'Beschriften                            [2696]
710 '                                       [09BE]
720 LOCATE 62,3:PRINT"X(Laenge)=":LOCATE
     62,4:PRINT"Y(Hoehe) ="                 [D6D4]
730 CLS #1:CLS #2                           [BF42]
740 LOCATE 2,2:PRINT"X Env-Composer "+CH
    R$(164)+" 1986 by<2>Michael Wessel X
    "                                       [CAB8]
750 LOCATE 4,23:PRINT"X VOLUME X"+STRING
    $(66,143)                               [C1A6]
760 |M1                                     [346E]
770 LOCATE 8,23:PRINT"X<== Zeit in 0.01
    Sek . ==>X"                             [8A96]
780 '                                       [B2CC]
790 'Gitter zeichnen                        [4B94]
800 '                                       [10BE]
810 |M2                                     [6B68]
820 |XON                                    [1056]
830 WINDOW #3,79,79,2,24:PAPER #3,0         [686A]
840 LOCATE #3,1,5:PRINT #3,"PAGE:"+HEX$(
    page)                                   [399C]
850 ORIGIN 0,0                              [3962]
860 FOR ix=  4*8 TO 77*8 STEP   8:MOVE i
    x,4*16-1:DRAW ix,20*16-1,1:NEXT         [9ABA]
870 FOR iy=20*16 TO 4*16 STEP -16:MOVE 7
    7*8,iy-1:DRAW 4*8,iy-1,1:  NEXT         [A2FA]
880 MOVE 77*8,20*16:DRAW 3*8,20*16:MOVE
    77*8,4*16-1:DRAW 77*8,16                [79BA]
890 INK 1,0:IF page>0 THEN PAPER #3,1:GO
    TO 1330                                 [35B0]
900 '                                       [0EC0]
910 'Env-Nummer eingeben                    [4110]
920 '                                       [10C4]
930 enves=0                                 [4D94]
940 FOR i=0 TO 14                           [46C4]
```

```
950 IF l(0,i)<>0 THEN enves=enves+1                    [A6C6]
960 NEXT                                               [1DFC]
970 text$=STR$(enves)+" = Anzahl der bel
    egten Huellkurven ."                               [0906]
980 GOSUB 2410                                         [C050]
990 |WAIT,&5000                                        [FF14]
1000 |M1                                               [50B6]
1010 INPUT #5,"Env Nummer :",n                         [1E16]
1020 IF n<1 OR n>15 THEN 1010                          [D222]
1030 n=n-1                                             [AEB6]
1040 FOR i=0 TO 4                                      [E9B2]
1050 IF l(i,n)=0 THEN NEXT:GOTO 1160                   [3996]
1060 text$=" Diese Huellkurve existiert
     bereits !"                                        [F1E4]
1070 GOSUB 2410                                        [599E]
1080 |WAIT,&5000                                       [5262]
1090 |M2                                               [73CA]
1100 INPUT #7," Soll sie geloescht werde
     n <J> <N> ";wahl$                                 [F9E2]
1110 IF UPPER$(wahl$)<>"J" THEN 1000                   [8836]
1120 FOR i=0 TO 4:t(i,n)=0:s(i,n)=0:l(i,
     n)=0:NEXT                                         [C52A]
1130 text$=" O.K.,Huellkurve geloescht .
     ":GOSUB 2410                                      [920A]
1140 enves=enves-1                                     [C286]
1150 |WAIT,&5000                                       [2B5E]
1160 |M0                                               [62C2]
1170 LOCATE 52,1:PRINT"ENV":LOCATE 52,2:
     PRINT USING"##";n+1                               [9D98]
1180 text$=" Start-Lautstaerke der Envel
     ope setzen ."                                     [B1E8]
1190 GOSUB 2410                                        [40A4]
1200 LOCATE #3,1,14:PRINT #3,"ENVES:"+HE
     X$(enves);                                        [5F90]
1210 PAPER #3,1                                        [3B5E]
1220 GOTO 1370                                         [A312]
1230 '                                                 [B81A]
1240 'Cursor-Routine                                   [6D3E]
1250 '                                                 [921E]
1260 |XON                                              [3EB4]
1270 a$=INKEY$:IF a$="" THEN 1270                      [35F0]
1280 IF a$=CHR$(13) THEN 3230                          [9A92]
1290 LOCATE x+5,y:PEN 0:PRINT CHR$(233)                [00FE]
1300 IF INKEY(0)=0 THEN y=y-1 ELSE IF IN
     KEY(2)=0 THEN y=y+1                               [E4FE]
1310 IF INKEY(8)=0 THEN 2590 ELSE IF ste
     igung<255 AND INKEY(1)=0 THEN 2670                [A4BE]
1320 IF INKEY(47)=0 AND part>-1 THEN 248
     0                                                 [25C2]
1330 IF x+ofs<ax+1 THEN x=ax+1-ofs                     [0480]
1340 IF y<6 THEN y=6 ELSE IF y>21 THEN y
     =21                                               [942E]
1350 IF x>72 THEN x=72 ELSE IF x<0 OR pa
     rt<0 THEN x=0                                     [B430]
1360 IF INKEY(9)=0 AND x<72 THEN IF part
     <0 THEN ay=y:part=0:erlau(n)=16-(y-
     5):CLS #7:x=x+1:GOTO 1370 ELSE IF p
     art<5 THEN 1460                                   [EE3C]
1370 PEN 1:LOCATE x+5,y:PRINT CHR$(233)                [D1FE]
1380 IF ay<>y THEN steigung=((x+ofs-ax)/
     ABS(y-ay)) ELSE steigung=x+ofs-ax                 [AFC0]
1390 IF steigung=INT(steigung) THEN PAPE
     R #1,1:PEN #1,0 ELSE PAPER #1,0:PEN
      #1,1                                             [F34A]
1400 CLS #1:IF x<72 THEN PRINT #1,INT(st
     eigung);                                          [3C74]
1410 CLS #2:PRINT #2,STR$(16-(y-5));                   [EE1C]
1420 GOTO 1270                                         [5514]
1430 '                                                 [B81E]
1440 'Steigung o. Gefaelle berechnen                   [1824]
1450 '                                                 [9622]
1460 dx=x+ofs-ax:dy=ABS(y-ay)                          [BEE0]
1470 IF (dx<>0 AND dy<>0) THEN steigung=
     (dx/dy) ELSE steigung=0                           [F026]
1480 IF (dx/csteig)<1 THEN text$=" Schri
     tt zu klein !!!":GOSUB 2410:GOTO 13
     70                                                [3A04]
1490 IF steigung<>INT(steigung) THEN GOT
     O 1370                                            [A864]
1500 PEN 1                                             [DE34]
1510 IF y>ay THEN 1660                                 [29C6]
1520 '                                                 [B91E]
1530 'Steigung (Positiv)                               [326A]
1540 '                                                 [9322]
1550 IF ax>ofs THEN WINDOW #4,ax+5-ofs,x
     +5,ay,21 ELSE WINDOW #4,5,x+5,ay,21
                                                       [52B8]
1560 xx=x                                              [0C22]
1570 FOR iy=y TO ay                                    [4592]
1580 IF xx+5>4 THEN WINDOW #3,xx+5,x+5,i
     y,iy ELSE WINDOW #3,5,x+5,iy,iy                   [9AE0]
1590 CLS #3                                            [F48E]
1600 xx=xx-steigung                                    [072E]
1610 NEXT                                              [594E]
1620 GOTO 1760                                         [7E20]
1630 '                                                 [C022]
1640 'Gefaelle (Negativ)                               [CBCC]
1650 '                                                 [E226]
1660 IF ax>ofs THEN WINDOW #4,ax+5-ofs,x
     +5,y,21 ELSE WINDOW #4,5,x+5,y,21                 [3538]
1670 xx=x                                              [4726]
1680 IF ax>=ofs THEN ddx=ax-ofs ELSE ddx
     =0                                                [5614]
1690 FOR iy=y TO ay STEP -1                            [F94C]
1700 IF xx+5>4 THEN WINDOW #3,ddx+5,xx+5
     ,iy,iy:CLS #3                                     [0D18]
```

```
1710 xx=xx-steigung                                    [F632]
1720 NEXT                                              [F052]
1730 '                                                 [9824]
1740 'in Arrays setzen                                 [A26A]
1750 '                                                 [9628]
1760 CLS #5                                            [F690]
1770 PAPER #4,1:PEN #4,0:CLS #4                        [517C]
1780 |M1                                               [6ED4]
1790 PRINT #5,"P"part+1":X";                           [8534]
1800 IF ay=y THEN PRINT #5," * HOLD * X"
     :GOTO 1830                                        [34FC]
1810 PRINT #5,dy;:IF ay>y THEN PRINT #5,
     "+S "; ELSE PRINT #5,"-S ";                       [2246]
1820 PRINT #5,"&"+HEX$(steigung)+" X";                 [24A6]
1830 IF ay=y THEN t(part,n)=1:s(part,n)=
     16:l(part,n)=ROUND(dx/csteig):GOTO
     1870                                              [42B8]
1840 IF y>ay THEN s(part,n)=-1 ELSE s(pa
     rt,n)=1                                           [B070]
1850 t(part,n)=dy                                      [3C3C]
1860 l(part,n)=steigung                                [6D40]
1870 text$=" ENV"+STR$(n+1)                            [E88C]
1880 FOR i=0 TO 4                                      [81CA]
1890 text$=text$+","+STR$(t(i,n))+","+ST
     R$(s(i,n))+","+STR$(l(i,n)):NEXT                  [B0CA]
1900 GOSUB 2410                                        [70A2]
1910 ax=x+ofs:ay=y                                     [FB6C]
1920 part=part+1                                       [37E6]
1930 le(n)=0:FOR i=0 TO 4:le(n)=le(n)+l(
     i,n)*t(i,n):NEXT                                  [D60A]
1940 GOTO 1330                                         [731C]
1950 '                                                 [C22C]
1960 'Env spielen                                      [B7A0]
1970 '                                                 [E430]
1980 |SVSCR,svscr                                      [8CB8]
1990 WINDOW #4,5,77,6,21:PAPER #4,0:PEN
     #4,1:CLS #4                                       [846E]
2000 |XOFF:|M0:PRINT #4:PRINT#4,"Env Nr.
     :"n+1                                             [38D6]
2010 |M1:PRINT #4,"<4>Lautstaerke :"                   [59D4]
2020 PRINT #4                                          [E4D0]
2030 PRINT #4:FOR i=0 TO 4:PRINT #4,"Par
     t"i+1":";                                         [A0B4]
2040 PRINT #4,USING"##";t(i,n);:PRINT #4
     ,USING"###";s(i,n);:PRINT #4," ";:P
     RINT #4,USING"###";l(i,n):NEXT                    [6FB0]
2050 ENV n+1,t(0,n),s(0,n),l(0,n),t(1,n)
     ,s(1,n),l(1,n),t(2,n),s(2,n),l(2,n)
     ,t(3,n),s(3,n),l(3,n),t(4,n),s(4,n)
     ,l(4,n)                                           [1B40]
2060 PRINT #4:PRINT #4:PRINT #4,"<TASTE>
      druecken zum verlassen ."                        [5702]
2070 IF INKEY$<>"" THEN |M2:|SORESET:|LD
     SCR,svscr:GOTO 1260                               [9F4C]
2080 LOCATE #4,18,3:PRINT #4,PEEK(&B56B)
                                                       [F504]
2090 |SOTEST,1:IF PEEK(&5000)=4 THEN SOU
     ND 1,284,le(n),erlau(n),n+1                       [BE54]
2100 GOTO 2070                                         [A30A]
2110 '                                                 [BD16]
2120 'Sichere Huellkurven                              [D408]
2130 '                                                 [BB1A]
2140 |SVSCR,svscr:WINDOW #4,5,77,6,21:PA
     PER #4,0:PEN #4,1:CLS #4                          [9F9E]
2150 PRINT #4:PRINT #4                                 [7954]
2160 |M1:INPUT #4," Erste Zeile<3>:",erz
     e                                                 [7D98]
2170 INPUT #4," Zeilenabstand :",zeab                  [BFDA]
2180 |M2:PRINT #4:PRINT #4,"<3>X Generie
     rtes BASIC-Programm : X"                          [30DA]
2190 PRINT #4                                          [B5E0]
2200 IF erze=0 THEN erze=10                            [8DF2]
2210 IF zeab=0 THEN zeab=10                            [89A4]
2220 OPENOUT"!env-bl.bas"                              [693A]
2230 FOR i=0 TO 14                                     [5B18]
2240 IF l(0,i)=0 THEN 2320                             [6778]
2250 zeile$=MID$(STR$(erze),2)+" ENV "+M
     ID$(STR$(i+1),2)                                  [9BAE]
2260 FOR ii=0 TO 4                                     [A88E]
2270 zeile$=zeile$+","+MID$(STR$(t(ii,i)
     ),2)+","+MID$(STR$(s(ii,i)),1)+","+
     MID$(STR$(l(ii,i)),2)                             [147E]
2280 NEXT ii                                           [893A]
2290 zeile$=zeile$+" ' SOUND x,x,"+MID$(
     STR$(le(i)),2)+","+MID$(STR$(erlau(
     i)),2)+","+MID$(STR$(i+1),2)+",x,x"
                                                       [319A]
2300 PRINT #4," "+zeile$:PRINT #9,zeile$
                                                       [BF1A]
2310 erze=erze+zeab                                    [9DB8]
2320 NEXT i                                            [DE5E]
2330 CLOSEOUT                                          [66AC]
2340 PRINT #4:PRINT #4,"<3>Fertig.Das Pr
     ogramm ist unter dem Namen":PRINT #
     4,"<3>ENV-BL.BAS gesichert ."                     [2F18]
2350 GOSUB 3570                                        [CDB2]
2360 |LDSCR,svscr                                      [9C78]
2370 GOTO 1260                                         [2C1C]
2380 '                                                 [9828]
2390 'Subroutine : Textmeldungen in Wind
     ow #7                                             [6458]
2400 '                                                 [921A]
2410 |M2:CLS #7                                        [C1F0]
```

Listing 1. »Envelope Composer«-Hauptprogramm

```
2420 PRINT #7,text$                          [9108]
2430 FOR i=7 TO 0 STEP -1:SOUND 1,284,3,
     i:NEXT                                  [D60A]
2440 RETURN                                  [AD94]
2450 '                                       [9524]
2460 'Page + 1                               [9358]
2470 '                                       [B728]
2480 IF ay<>y THEN steigung=INT((x+ofs-a
     x+72)/ABS(y-ay)) ELSE steigung=x+of
     s-ax+72                                 [5BEA]
2490 IF steigung>255 THEN 1370               [8F30]
2500 ofs=ofs+72                              [FE90]
2510 page=page+1                             [1176]
2520 x=0                                     [EB9C]
2530 WINDOW #4,5,77,6,21:PAPER #4,0:PEN
     #4,1                                    [B836]
2540 CLS #4:PAPER #4,1:PEN #4,0              [A374]
2550 GOTO 810                                [71BC]
2560 '                                       [9628]
2570 'Naechste Position fuer Eingabe auf
      Y-Hoehe berechnen -                    [E7A8]
2580 '                                       [BC2C]
2590 IF y=ay THEN x=x-1:GOTO 1330            [6B06]
2600 FOR ix=x+ofs-1 TO ax+1 STEP-1           [5C50]
2610 dx=ix-ax:dy=ABS(y-ay)                   [FDC8]
2620 IF dx/dy=INT(dx/dy) AND dx/dy<256 T
     HEN x=ix-ofs ELSE NEXT                  [4AB0]
2630 GOTO 1330                               [AF16]
2640 '                                       [BA26]
2650 'Naechste Position fuer Eingabe auf
      Y-Hoehe berechnen +                    [C5A2]
2660 '                                       [B82A]
2670 IF y=ay THEN x=x+1:GOTO 1330            [2A00]
2680 FOR ix=x+ofs+1 TO ofs+x+72              [D97C]
2690 dx=ix-ax:dy=ABS(y-ay)                   [E5D8]
2700 IF dx/dy=INT(dx/dy) THEN x=ix-ofs E
     LSE NEXT                                [A906]
2710 GOTO 1330                               [4914]
2720 '                                       [BA24]
2730 'Huellkurvendiagramm ausgeben (Uebe
     rsicht)                                 [5CA6]
2740 '                                       [9428]
2750 |SVSCR,svscr                            [8AB0]
2760 |XOFF                                   [DB3C]
2770 WINDOW #4,1,80,1,25                     [0040]
2780 CLS                                     [20A6]
2790 |M0:PRINT "Diagramm von"                [EE56]
2800 PRINT "Huellkurve Nr. "n+1              [10F4]
2810 |M2                                     [76CC]
2820 ORIGIN 0,300                            [C386]
2830 iy=0:xphys=0:yphys=erlau(n)             [6356]
2840 FOR i=-50 TO 200 STEP 50:PLOT 0,-i:
     DRAW 637,-i:NEXT                        [3CE2]
2850 MOVE 0,0:DRAWR 0,-200:DRAWR 0,250:M
     OVE 637,0:DRAWR 0,-200:DRAWR 0,250      [59C6]
2860 MOVE 0,erlau(n)*3+1                     [4068]
2870 FOR i=0 TO 4                            [37CA]
2880 IF s(i,n)<>ABS(s(i,n)) THEN yvektor
     =-t(i,n) ELSE yvektor=t(i,n)            [D50E]
2890 IF s(i,n)=16 THEN yvektor=0             [D0FA]
2900 xphys=xphys+l(i,n)*2                    [6ECE]
2910 IF xphys<637 THEN DRAWR l(i,n)*2,yv
     ektor*3:yphys=yphys+yvektor ELSE iy
     =iy-50:ORIGIN 0,300+iy:MOVE 0,yphys
     *3:xphys=0:GOTO 2900                    [C21A]
2920 NEXT                                    [FE58]
2930 |M1:LOCATE 1,20:INPUT "Hardcopy <J>
      <N> ";wahl$:IF UPPER$(wahl$)="J" T
     HEN INPUT "<D>iagramm oder <B>ildsc
     hirm ";wahl$:|M2:IF UPPER$(wahl$)="
     B" THEN |LDSCR,svscr:|HARDCOPY ELSE
      |HARDCOPY                              [3DDA]
2940 LOCATE #4,1,22:GOSUB 3570               [1168]
2950 |LDSCR,svscr                            [2282]
2960 GOTO 1260                               [A926]
2970 '                                       [C732]
2980 'Helppage                               [9B80]
2990 '                                       [C536]
3000 |SVSCR,svscr                            [8D9A]
3010 WINDOW #4,5,77,6,21:PAPER #4,0:PEN
     #4,1                                    [152A]
3020 CLS #4                                  [1B7C]
3030 |M0:PRINT #4,"X Help-page X"            [EEA0]
3040 |M1:PRINT #4,"X Direkteingaben<11>X
     "                                       [C4B6]
3050 |M2:PRINT #4," Cursorkeys <";:FOR i
     =240 TO 243:PRINT #4,CHR$(i);:NEXT:
     PRINT #4,"> = entsprechende Cursor-
     Richtung"                               [2968]
3060 PRINT #4," 2*<ESC> = Programm verla
     ssen"                                   [DFC6]
3070 PRINT #4," <COPY><2>= Huellkurvenpa
     rt definieren"                          [3CAA]
3080 PRINT #4," <ENTER> = Hauptmenue ein
     blenden"                                [0D88]
3090 PRINT #4," <SPACE> = eine Seite wei
     terblaettern"                           [C21E]
3100 |M1:PRINT #4,"X Hauptmenuekommandos
     <6>X":|M2                               [5B6C]
3110 PRINT #4," Helppage = die Helppage
     wird eingebendet"                       [B408]
3120 PRINT #4," Diagramm = Huellkurve wi
     rd in Diagrammform ausgeben"            [92DA]
3130 PRINT #4," Ende<5>= neue Huellkurve
      definieren , alte abschliessen"        [AC8E]
3140 PRINT #4," Spielen<2>= die momentan
     e Huellkurve wird hoerbar"              [243E]
3150 PRINT #4," Sichern<2>= alle definie
     rten Huellkurven auf Diskette/Kasse
     tte sichern"                            [333C]
3160 PRINT #4," <ENTER><2>= Funktion aus
     fuehren"                                [456E]
3170 GOSUB 3570                              [61B4]
3180 |LDSCR,svscr                            [167A]
3190 GOTO 1260                               [AB1E]
3200 '                                       [9318]
3210 'Pull-down-Menue                        [A66C]
3220 '                                       [911C]
3230 iy=3                                    [8174]
3240 |XOFF                                   [E430]
3250 |SVSCR,svscr                            [AFA8]
3260 WINDOW #4,5,77,6,21:PAPER #4,0:PEN
     #4,1                                    [0238]
3270 CLS #4                                  [F98A]
3280 |M0:PRINT #4,"XHaupt-MenueX"            [0EDA]
3290 |M1:PRINT #4:PRINT #4,"<2>Spielen"      [F2C8]
3300 PRINT #4,"<2>Sichern"                   [81CC]
3310 PRINT #4,"<2>Helppage"                  [9282]
3320 PRINT #4,"<2>Diagramm"                  [D67C]
3330 PRINT #4,"<2>Ende"                      [8F32]
3340 PRINT #4,"<2>Haupt-Menue verlassen"
                                             [5374]
3350 PRINT #4,"<2>Programm<4>verlassen"      [9B6E]
3360 GOTO 3430                               [A31E]
3370 a$=INKEY$:IF a$="" THEN 3370            [38FC]
3380 LOCATE #4,1,iy:PRINT #4,"<2>"           [09EC]
3390 IF a$=CHR$(240) THEN iy=iy-1            [552C]
3400 IF a$=CHR$(241) THEN iy=iy+1            [441A]
3410 IF a$=CHR$(13) THEN 3450                [EC94]
3420 IF iy<3 THEN iy=3 ELSE IF iy>9 THEN
      iy=9                                   [CBC4]
3430 LOCATE #4,1,iy:PRINT #4.CHR$(247)+C
     HR$(246)                                [0DEC]
3440 GOTO 3370                               [B922]
3450 |M2:|LDSCR,svscr:ON iy-2 GOTO 1980,
     2140,3000,2750,3490,1260,3530           [9A72]
3460 '                                       [E528]
3470 'Neue Huellkurve definieren             [0B44]
3480 '                                       [972C]
3490 MODE 2:GOTO 200                         [1318]
3500 '                                       [BD1E]
3510 'Programm verlassen                     [8E90]
3520 '                                       [BB22]
3530 INK 1,0:PEN 1:MODE 2:END                [9454]
3540 '                                       [9926]
3550 'Auf Taste warten                       [8104]
3560 '                                       [972A]
3570 |M1:PRINT #4                            [014E]
3580 PRINT #4,"<TASTE> druecken zum verl
     assen ."                                [3A1A]
3590 CALL &BB06:|M2:RETURN                   [0C18]
```

Listing 1. »Envelope Composer«-Hauptprogramm (Schluß)

```
100 '**************************************  [31D4]
101 '* ENVELOPE.BIN - DATA-Lader von CPC *  [E4AE]
102 '**************************************  [A3D8]
103 '                                        [DEB6]
104 DATA 5000,84,01,0A,50,21,5E,50,C3,4613  [52D2]
105 DATA 5008,D1,BC,2D,50,C3,62,50,C3,40D3  [CD64]
106 DATA 5010,74,50,C3,86,50,C3,93,50,3EFA  [A444]
107 DATA 5018,C3,97,50,C3,9D,50,C3,FB,46A5  [629A]
108 DATA 5020,50,C3,A3,50,C3,B3,50,C3,0D17  [1838]
109 DATA 5028,CD,50,C3,1B,A0,53,56,53,6FE3  [5E88]
110 DATA 5030,43,D2,4C,44,53,43,D2,53,1AA3  [D71E]
111 DATA 5038,4F,54,45,53,D4,53,4F,52,3830  [3718]
112 DATA 5040,45,53,45,D4,58,4F,CE,58,3198  [694C]
113 DATA 5048,4F,46,C6,57,41,49,D4,4D,2979  [1164]
114 DATA 5050,B0,4D,B1,4D,B2,48,41,52,5DD0  [6768]
115 DATA 5058,44,43,4F,50,D9,00,53,50,381E  [4510]
116 DATA 5060,0A,50,FE,01,C0,DD,56,01,0B09  [605A]
117 DATA 5068,DD,5E,00,21,00,C0,01,FF,78ED  [0D9E]
118 DATA 5070,3F,ED,B0,C9,FE,01,C0,DD,38F9  [B01A]
119 DATA 5078,66,01,DD,6E,00,11,00,C0,2E84  [B83A]
120 DATA 5080,01,FF,3F,ED,B0,C9,FE,01,3129  [26BC]
121 DATA 5088,C0,DD,7E,00,CD,AD,BC,32,5D16  [3CF8]
122 DATA 5090,00,50,C9,CD,A7,BC,C9,3E,0794  [04AC]
123 DATA 5098,01,CD,9F,BB,C9,3E,00,CD,2DED  [542E]
124 DATA 50A0,9F,BB,C9,21,C8,B1,36,00,7E98  [AAB2]
125 DATA 50A8,21,CF,B1,36,F0,21,D0,B1,3015  [935A]
126 DATA 50B0,36,0F,C9,21,C8,B1,36,01,0719  [A83A]
127 DATA 50B8,21,CF,B1,36,C0,21,D0,B1,3195  [446C]
128 DATA 50C0,36,30,21,D1,B1,36,0C,21,1B59  [6F20]
129 DATA 50C8,D2,B1,36,03,C9,21,C8,B1,445D  [D78A]
130 DATA 50D0,36,02,21,CF,B1,36,80,21,1721  [F0F8]
131 DATA 50D8,D0,B1,36,40,21,D1,B1,36,4598  [F53C]
132 DATA 50E0,20,21,D2,B1,36,10,21,D3,0871  [6CF0]
133 DATA 50E8,B1,36,08,21,D4,B1,36,04,521C  [AC32]
134 DATA 50F0,21,D5,B1,36,02,21,D6,B1,3109  [8820]
135 DATA 50F8,36,01,C9,FE,01,C0,06,05,0E81  [2860]
136 DATA 5100,DD,66,01,DD,6E,00,2B,7D,79AB  [F7C0]
137 DATA 5108,FE,00,C2,06,51,7C,FE,00,65A4  [5E80]
```

```
138 DATA 5110,C2,06,51,10,EB,C9,00,00,6FDC  [1F62]
139 DATA *ENDE*                              [DECE]
140 adr=&5000:zeile=104:MEMORY adr-1         [5F02]
141 READ d$:IF d$="*ENDE*"THEN 152           [7E86]
142 pr=0                                     [610C]
143 FOR i=1 TO 8                             [3062]
144 READ a$:a=VAL("&"+a$)                    [A540]
145 POKE adr,a:adr=adr+1                     [B81C]
146 pr=pr*2:IF pr>65535 THEN pr=pr-65535     [E19C]
147 pr=UNT(pr)XOR a:IF pr<0 THEN pr=pr+6553
    6                                        [A4B4]
148 NEXT i                                   [9A0A]
149 READ pr$:pr2=VAL("&"+pr$):IF pr2<0 THEN
     pr2=pr2+65536                           [2A96]
150 IF pr<>pr2 THEN PRINT"Pruefsummenfehler
     in Zeile";zeile:STOP                    [C60C]
151 zeile=zeile+1:GOTO 141                   [DB56]
152 SAVE"ENVELOPE.BIN",B,&5000,&116:END      [C420]
```

Listing 2. Der DATA-Lader enthält den Maschinencode für die Befehlserweiterung zum Envelope Composer

```
100 '************************************    [31D4]
101 '* ENVELOPE.HRD - DATA-Lader von CPC *   [58B8]
102 '************************************    [A3D8]
103 '                                         [DEB6]
104 DATA A000,01,09,A0,21,17,A0,C3,D1,17BF  [913C]
105 DATA A008,BC,0E,A0,C3,1B,A0,48,41,4739  [4184]
106 DATA A010,52,44,43,4F,50,D9,00,00,3574  [96E8]
107 DATA A018,00,00,00,CD,0B,BC,22,4C,0E70  [5E5A]
108 DATA A020,A1,CD,11,BC,32,4E,A1,21,6BEB  [26C8]
109 DATA A028,5F,A1,3D,28,09,21,57,A1,0223  [8534]
110 DATA A030,3D,28,03,21,63,A1,01,08,1766  [EBD6]
111 DATA A038,00,11,4F,A1,ED,B0,ED,73,03B1  [388E]
112 DATA A040,6E,A1,3A,4E,A1,87,FE,01,1A09  [BBA2]
113 DATA A048,CE,00,87,32,66,A1,0E,0D,7565  [5A64]
114 DATA A050,CD,37,A1,0E,0A,CD,37,A1,7C2B  [2BD8]
115 DATA A058,3E,02,CD,AD,A0,3E,7F,32,09C4  [51D8]
116 DATA A060,65,A1,2A,4C,A1,7C,F6,C0,1E94  [B0B4]
117 DATA A068,67,06,1D,05,20,05,3E,78,30E0  [DF22]
118 DATA A070,32,65,A1,04,C5,CD,C2,A0,1018  [2D5A]
119 DATA A078,3A,4E,A1,FE,02,CC,C2,A0,1744  [5ED2]
120 DATA A080,11,00,08,A7,ED,52,7C,F6,05DE  [AC90]
121 DATA A088,3F,3C,28,08,11,B0,3F,19,172F  [7D66]
122 DATA A090,7C,F6,F8,67,0E,0A,CD,37,1B05  [E8CA]
123 DATA A098,A1,0E,0D,CD,37,A1,E5,3E,5CB8  [7D00]
124 DATA A0A0,42,CD,1E,BB,C2,47,A1,E1,1C9F  [6E02]
125 DATA A0A8,C1,10,C0,3E,03,21,70,A1,7FBD  [5E94]
126 DATA A0B0,4E,23,06,00,09,3D,20,F8,2F04  [2B44]
127 DATA A0B8,46,23,4E,23,CD,37,A1,10,27D6  [A07E]
128 DATA A0C0,F9,C9,E5,21,70,A1,3A,4E,514E  [6CC4]
129 DATA A0C8,A1,FE,02,28,05,4E,23,06,6C90  [ED84]
130 DATA A0D0,00,09,CD,B8,A0,E1,3A,4E,16DE  [97DA]
131 DATA A0D8,A1,FE,02,06,28,28,02,06,6EC2  [CA72]
132 DATA A0E0,50,C5,E5,11,67,A1,3E,07,0537  [FC50]
133 DATA A0E8,ED,A0,2B,01,00,08,09,30,5BF2  [827A]
134 DATA A0F0,0A,01,50,C0,09,47,7C,E6,020A  [6856]
135 DATA A0F8,C7,67,78,3D,20,EA,21,4F,7435  [95A6]
136 DATA A100,A1,3A,66,A1,47,C5,11,67,59B9  [1262]
137 DATA A108,A1,06,07,1A,13,A6,FE,01,53BD  [2582]
138 DATA A110,3F,CB,11,10,F6,3A,65,A1,2953  [5A6C]
139 DATA A118,A1,4F,CD,37,A1,3A,4E,A1,5C4D  [49F8]
140 DATA A120,A7,CC,37,A1,C1,23,10,DD,6A09  [4E96]
141 DATA A128,E1,C1,7C,E6,F8,4F,23,7C,47E6  [6BE8]
142 DATA A130,E6,07,B1,67,10,AB,C9,79,6157  [826E]
143 DATA A138,CD,2B,BD,D8,C5,E5,3E,42,73E2  [5512]
144 DATA A140,CD,1E,BB,E1,C1,28,F0,ED,7FD5  [CF3E]
145 DATA A148,7B,6E,A1,C9,00,00,00,00,3EB0  [3954]
146 DATA A150,00,00,00,00,00,00,00,80,0080  [5C50]
147 DATA A158,40,20,10,08,04,02,01,88,2A22  [3ABC]
148 DATA A160,44,22,11,AA,55,00,00,00,20A8  [BCE0]
149 DATA A168,00,00,00,00,00,00,00,00,0000  [B548]
150 DATA A170,04,1B,4C,40,01,04,1B,4B,0925  [B80A]
151 DATA A178,40,01,03,1B,41,07,03,1B,2399  [4CE4]
152 DATA A180,41,0C,00,00,00,00,00,00,2380  [117A]
153 DATA *ENDE*                              [78C6]
154 adr=&A000:zeile=104:MEMORY adr-1         [7D24]
155 READ d$:IF d$="*ENDE*"THEN 166           [FB9A]
156 pr=0                                     [5316]
157 FOR i=1 TO 8                             [3A6C]
158 READ a$:a=VAL("&"+a$)                    [F64A]
159 POKE adr,a:adr=adr+1                     [1826]
160 pr=pr*2:IF pr>65535 THEN pr=pr-65535     [AB94]
161 pr=UNT(pr)XOR a:IF pr<0 THEN pr=pr+6553
    6                                        [7EAC]
162 NEXT i                                   [1002]
163 READ pr$:pr2=VAL("&"+pr$):IF pr2<0 THEN
     pr2=pr2+65536                           [948E]
164 IF pr<>pr2 THEN PRINT"Pruefsummenfehler
     in Zeile";zeile:STOP                    [5E16]
165 zeile=zeile+1:GOTO 155                   [226A]
166 SAVE"ENVELOPE.HRD",b,&A000,&182:END      [6392]
```

Listing 3. Die Hardcopyroutine läßt sich an fast alle Drucker anpassen oder gegen eigene Unterprogramme austauschen

Lustiges Boxenbasteln per Computer

Das Selberbauen von Lautsprecherboxen für die heimische Stereoanlage erspart eine Menge Geld. Doch das Berechnen der Bauteiledaten für die elektrische Frequenzweiche sowie das Ermitteln des Boxenvolumens und der äußeren Abmessungen ist nicht jedermanns Sache.

Sie glauben gar nicht, was Ihr Computer alles kann. Mit dem ausgetüftelten Programm »Boxenkit« ist er sogar in der Lage, Ihnen sämtliche elektrische und physikalische Daten für den Selbstbau von Lautsprecherboxen zu berechnen und den Schaltplan auszugeben.

Zudem können Sie die Werte für eine Lautsprecherbox beliebig oft mit wechselnden Parametern berechnen – bis Sie Ihre Traumbox gefunden haben. Im folgenden werden wir Ihnen exemplarisch einen kompletten Arbeitsgang mit Boxenkit vorstellen.

Zu Beginn des Programms erscheint das Hauptmenü, das folgende acht Punkte zeigt:

- Daten der Frequenzweiche berechnen
- Gehäusemaße berechnen
- Schaltplan anzeigen
- Daten speichern
- Daten lesen
- Daten drucken
- Programm erläutern
- Programm beenden

Wenn Sie schon ein passendes Lautsprecherset besitzen, wählen Sie den ersten Punkt des Menüs an.

Unter der Kopfzeile des Menüpunktes »Frequenzweichen« erscheint das entsprechende Arbeitsblatt. Sie können zwischen einer Zweiweg- und einer Dreiwegweiche wählen. Haben Sie sich entschieden, so müssen Sie die Impedanzen (Scheinwiderstände) der einzelnen Lautsprecher sowie die Übergangsfrequenzen zwischen den Frequenzbereichen der Lautsprecher eingeben.

Danach berechnet der Computer in Sekundenschnelle die Daten für eine Frequenzweiche mit einer Flankensteilheit von 6 und 12 Dezibel pro Oktave (»6db/Okt.« beziehungsweise »12db/Okt.«). Die Werte werden in Form von zwei Tabellen ausgegeben.

Am unteren rechten Bildschirmrand erfolgt nun die Abfrage, ob die Werte mit anderen Daten erneut berechnet werden sollen, oder ob das Hauptmenü gewünscht ist.

Für die Berechnung des Boxenvolumens und der Gehäusemaße wählen Sie den zweiten Menüpunkt an. Wenn Sie

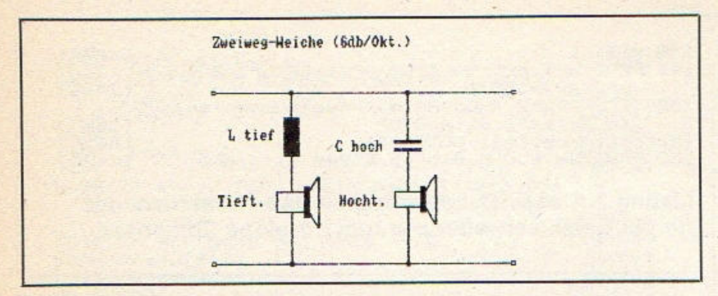


Bild 1. Der Schaltplan der Zweiweg-Frequenzweiche mit 6 Dezibel Flankensteilheit ...

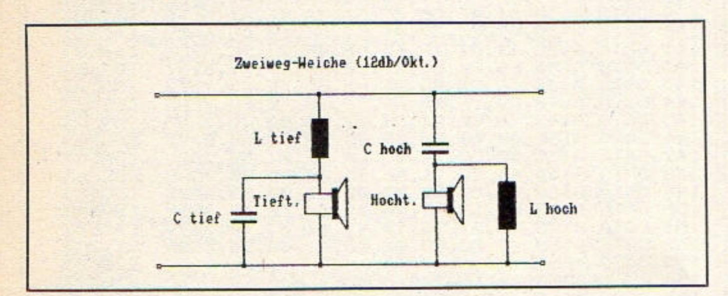


Bild 2. ... und mit 12 Dezibel Flankensteilheit

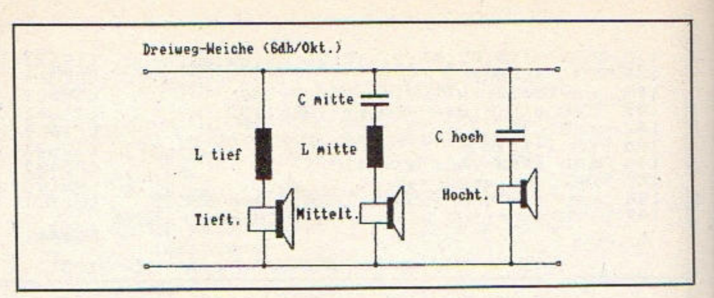


Bild 3. Ebenso der Schaltplan der Dreiweg-Frequenzweiche für eine Flankensteilheit von 6 Dezibel ...

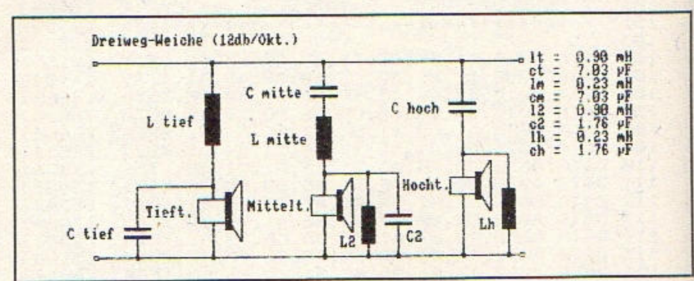


Bild 4. ... und 12 Dezibel (diesmal mit Bestückungsplan)

sich in diesem Metier nicht auskennen, sollten Sie sich jedoch zuvor mit dem vorletzten Menüpunkt Klarheit darüber verschaffen, worum es beim Berechnen der Boxenausmaße geht.

Innerhalb gewisser Grenzen kann der Anwender nun »seine« Lautsprecherbox entwerfen. Die Grundform der Box ist jedoch als Quader festgelegt, und Fehleingaben sowie Überschreitungen von Grenzwerten fängt Boxenkit ab.

Hat man sich dann für ein bestimmtes Verhältnis zwischen Höhe und Tiefe oder Höhe und Breite der Box entschieden, so berechnet der Computer den dritten Wert und gibt gleich noch die Zuschnittmaße für die benötigten Holzplatten aus, wobei die Materialstärke selbstverständlich berücksichtigt wird.

Darauf ist der dritte Menüpunkt an der Reihe, denn schließlich möchte man auch den Schaltplan für die Frequenzweiche sehen. Aus Platzgründen haben wir die Bilddateien der vier Schaltpläne, aus denen der Anwender auswählen kann und die vom Programm automatisch mit einem Bestückungsplan versehen werden, nicht abgedruckt. Der ambitionierte Leser kann die Schaltbilder jedoch aus Bild 1 bis 4 entnehmen und diese als Grafiken selbst programmieren oder aber die Leserservice-Diskette erwerben, auf der die vier Schaltpläne als für Boxenkit erkennbare Bilddateien (»2WEG6DB.GRF«, »2WEG12DB.GRF«, »3WEG6DB.GRF« und »3WEG12DB.GRF«) gespeichert sind.

Von den Schaltplänen lassen sich auch Hardcopies auf Druckern, die mit dem Epson-Steuerzeichensatz arbeiten, anfertigen.

Sämtliche Daten können in den zwei folgenden Menüpunkten auf Datenträger geschrieben und von Datenträger gelesen werden. Die Dateien werden dabei jeweils mit der Extension »box« versehen.

Zu guter Letzt kann man sich die Boxen-Daten über den Menüpunkt »Daten drucken« fein säuberlich auf einen Bogen im DIN-A4-Format ausgeben lassen.

Mit diesem Einkaufszettel gehen Sie dann zum Elektronikshop und zum Schreiner, um das Material für Ihre Box(en) zu besorgen. Nichts klingt so gut wie die ersten Töne aus einer selbstgebauten Lautsprecherbox!

(Thomas H. Richter/ma)

Steckbrief	
Programm:	Boxenkit
Computer:	CPC 464/664/6128
Checksummer:	Explora
Datenträger:	Kassette, Diskette

```
10   '******************************     [0B68]
20   '**  Boxenberechnung V.4.3   **     [D9CE]
30   '******************************     [8F6C]
40   '**      fuer CPC 464        **     [1E32]
50   '**      fuer CPC 664        **     [7238]
60   '**      fuer CPC 6128       **     [E45C]
70   '******************************     [AF74]
80   '**          von             **     [5254]
90   '**    Thomas  H. Richter    **     [A516]
100  '******************************     [2B88]
110  '** (C) 1986 by  G A L A C O **     [B99E]
120  '******************************     [AD8C]
130  '**        Erster  Teil      **     [7D0C]
140  '******************************     [4790]
150  '**      Installationen &    **     [87A0]
160  '**         Hauptmenue;      **     [3B7A]
170  '**   Boxengehaeuse geschl.  **     [245C]
180  '******************************     [5398]
190 CLEAR:IF HIMEM>&9FFF THEN MEMORY &9F
    FF:GOSUB 3950                        [4F5A]
200 MODE 1:BORDER 27:INK 0,27:INK 1,1:au
    sgabe=0:ffff=0                       [7C64]
210 ff=0                                 [77D8]
220 PAPER 0:PEN 1                        [27D4]
230 GOSUB 5640                           [2F48]
240 LOCATE 9,6:PRINT CHR$(24);STRING$(22
    ," ");CHR$(24)                       [B16C]
250 LOCATE 9,7:PRINT CHR$(24);" HIFI-BOX
    ENBERECHNUNG ";CHR$(24)              [8C4C]
260 LOCATE 9,8:PRINT CHR$(24);STRING$(22
    ," ");CHR$(24)                       [D574]
270 LOCATE 10,11:PRINT"Frequenzweichen..
    ..1"                                 [464E]
280 LOCATE 10,12:PRINT"Geschl. Gehaeuse.
    ..2"                                 [F128]
290 LOCATE 10,13:PRINT"Schaltplan zeigen
    ..3"                                 [E794]
300 LOCATE 10,14:PRINT"Daten speichern..
    ..4"                                 [8D82]
310 LOCATE 10,15:PRINT"Daten einlesen...
    ..5"                                 [E908]
320 LOCATE 10,16:PRINT"Daten ausdrucken.
    ..0"                                 [8E0E]
330 LOCATE 10,18:PRINT"Erklaerungen.....
    ..?"                                 [15E6]
340 LOCATE 10,19:PRINT"Programm beenden.
```

```
    ..6"                                  [210E]
350 LOCATE 10,23:PRINT"Ihre Wahl bitte..
    ..?";                                 [9556]
360 wahl$=UPPER$(INKEY$):IF wahl$="" THE
    N 360                                 [5F5A]
370 IF wahl$="3" THEN GOSUB 3610:GOTO 20
    0                                     [3AD6]
380 IF wahl$="4" THEN GOSUB 4340:GOTO 20
    0                                     [50DC]
390 IF wahl$="1" THEN ww(2)=1:GOTO 1180   [F930]
400 IF wahl$="0" THEN 2640                [EB3E]
410 IF wahl$="2" THEN 460                 [FBE0]
420 IF wahl$="?" THEN GOSUB 4220:GOTO 20
    0                                     [9FE2]
430 IF wahl$="5" THEN GOSUB 5010:GOTO 20
    0                                     [C2CC]
440 IF wahl$="6" THEN 4110:GOTO 200       [9C90]
450 GOTO 200                              [CD48]
460 ww(1)=1                               [C630]
470 MODE 2:GOSUB 5660:BORDER 27:INK 0,27
    :INK 1,1                              [B252]
480 LOCATE 45,4:PRINT CHR$(24);"<13>";CH
    R$(24)                                [D31C]
490 LOCATE 45,5:PRINT CHR$(24);" GEHAEUS
    E... ";CHR$(24)                       [E802]
500 LOCATE 45,6:PRINT CHR$(24);"<13>";CH
    R$(24)                                [B512]
510 LOCATE 1,5:INPUT"V(as) - Wert<3>: ",
    vas$                                  [3518]
520 IF LEN(vas$)>3 OR vas$=""THEN LOCATE
     1,5:PRINT STRING$(30+LEN(vas$)," ")
    ;:GOTO 510                            [AEA2]
530 IF ASC(LEFT$(vas$,1))<48 OR ASC(LEFT
    $(vas$,1))>57 THEN LOCATE 1,5:PRINT
    STRING$(30+LEN(vas$)," ");:GOTO 510   [F3D8]
540 IF vas$="0" THEN LOCATE 1,5:PRINT ST
    RING$(30," ");:GOTO 510               [71D4]
550 IF LEN(vas$)>=4 THEN LOCATE 1,5:PRIN
    T STRING$(30+LEN(vas$)," "):GOTO 510
                                          [794E]
560 IF RIGHT$(vas$,1)="." THEN LOCATE 1,
    5:PRINT STRING$(30+LEN(vas$)," ");:G
    OTO 510                               [FE06]
570 vas=VAL(vas$):IF vas<10 THEN LOCATE
    1,5:PRINT STRING$(35," "):GOTO 510    [93C4]
580 LOCATE 1,7:INPUT"Q(ts) - Wert<3>: ",
    qts$                                  [A062]
590 IF LEN(qts$)>4 OR qts$=""THEN LOCATE
     1,7:PRINT STRING$(30+LEN(qts$)," ")
    ;:GOTO 580                            [B718]
600 IF ASC(LEFT$(qts$,1))<48 OR ASC(LEFT
    $(qts$,1))>57 THEN LOCATE 1,7:PRINT
    STRING$(30+LEN(qts$)," ");:GOTO 580   [373A]
610 IF LEN(qts$)=3 THEN x$=LEFT$(qts$,2)
    :x$=RIGHT$(x$,1):IF x$<>"." THEN LOC
    ATE 1,7:PRINT STRING$(30+LEN(qts$),"
     "):GOTO 580                          [0C78]
620 IF qts$="0" THEN LOCATE 1,7:PRINT ST
    RING$(30," ");:GOTO 580               [3E00]
630 IF LEN(qts$)>=5 THEN LOCATE 1,7:PRIN
    T STRING$(30+LEN(qts$)," "):GOTO 580
                                          [8298]
640 IF RIGHT$(qts$,1)="." THEN LOCATE 1,
    7:PRINT STRING$(30+LEN(qts$)," ");:G
    OTO 580                               [154E]
650 qts=VAL(qts$):IF qts<0.01 OR qts>0.6
    9 THEN LOCATE 1,7:PRINT STRING$(35,"
     "):GOTO 580                          [2E6C]
660 LOCATE 1,9:PRINT"Q(tg) - Wert<3>: 0.
    707"                                  [76EE]
670 qtg=0.707                             [E984]
680 vga=(qtg^2/qts^2)-1                   [8AB6]
690 vgb=vas/vga                           [A7E4]
700 vg=vgb*0.92                           [4406]
710 LOCATE 1,11:PRINT CHR$(24);"Volumen
    bedaempft : ";ROUND(vg,2);" Liter";C
    HR$(24)                               [70EC]
720 LOCATE 1,14:PRINT"Geben Sie nun die
    gewuenschten Abmessungen ein...":PRI
    NT"Die Materialstaerke darf zwischen
     08 und 99 mm liegen":PRINT"Aussenab
    messungen duerfen zwischen 15 und 20
    0 cm liegen.":LOCATE 60,24:PRINT"bit
    te <Taste>...":CALL &BB06             [F5E4]
730 FOR t=14 TO 16:LOCATE 1,t:PRINT STRI
    NG$(56," ");:NEXT:LOCATE 60,24:PRINT
     STRING$(19," ");                     [89B0]
740 LOCATE 1,14:INPUT"Materialstaerke in
     mm<2>: ",mast$                       [5D22]
750 IF LEN(mast$)>4 OR mast$=""THEN LOCA
    TE 1,14:PRINT STRING$(30+LEN(mast$),
    " ");:GOTO 740                        [1F9A]
760 IF ASC(LEFT$(mast$,1))<48 OR ASC(LEF
    T$(mast$,1))>57 THEN LOCATE 1,14:PRI
    NT STRING$(30+LEN(mast$)," ");:GOTO
    740                                   [BBCE]
770 IF LEN(mast$)=3 THEN x$=LEFT$(mast$,
    2):x$=RIGHT$(x$,1):IF x$<>"." THEN L
    OCATE 1,14:PRINT STRING$(30+LEN(mast
    $)," "):GOTO 740                      [A40C]
780 IF mast$="0" THEN LOCATE 1,14:PRINT
    STRING$(30," ");:GOTO 740             [EE20]
790 IF LEN(mast$)>=5 THEN LOCATE 1,14:PR
    INT STRING$(30+LEN(mast$)," "):GOTO
    740                                   [8972]
800 IF RIGHT$(mast$,1)="." THEN LOCATE 1
    ,14:PRINT STRING$(30+LEN(mast$)," ")
    ;:GOTO 740                            [1C16]
810 mast=VAL(mast$):IF mast=0 OR mast>10
    0 OR mast<8 THEN LOCATE 1,14:PRINT S
    TRING$(35," "):GOTO 740               [5C28]
820 LOCATE 1,16:INPUT"Standhoehe aussen
    in cm: ",auho$                        [BDA8]
830 IF LEN(auho$)>5 OR auho$=""THEN LOCA
    TE 1,16:PRINT STRING$(30+LEN(auho$),
    " ");:GOTO 820                        [B06C]
840 IF ASC(LEFT$(auho$,1))<48 OR ASC(LEF
    T$(auho$,1))>57 THEN LOCATE 1,16:PRI
    NT STRING$(30+LEN(auho$)," ");:GOTO
    820                                   [889E]
850 IF auho$="0" THEN LOCATE 1,16:PRINT
    STRING$(30," ");:GOTO 820             [580E]
860 IF LEN(auho$)>5 THEN LOCATE 1,16:PRI
    NT STRING$(30+LEN(auho$)," "):GOTO 8
    20                                    [6FD6]
870 IF RIGHT$(auho$,1)="." THEN LOCATE 1
    ,16:PRINT STRING$(30+LEN(auho$)," ")
    ;:GOTO 820                            [4006]
880 auho=VAL(auho$):IF auho=0 OR auho>20
    0 OR auho<15 THEN LOCATE 1,16:PRINT
    STRING$(35," "):GOTO 820              [3846]
890 LOCATE 1,18:INPUT"Tiefe oder Breite
    in cm: ",tibr$                        [28C2]
900 IF LEN(tibr$)>5 OR tibr$=""THEN LOCA
    TE 1,18:PRINT STRING$(30+LEN(tibr$),
    " ");:GOTO 890                        [D792]
910 IF ASC(LEFT$(tibr$,1))<48 OR ASC(LEF
    T$(tibr$,1))>57 THEN LOCATE 1,18:PRI
    NT STRING$(30+LEN(tibr$)," ");:GOTO
    890                                   [73C4]
920 IF tibr$="0" THEN LOCATE 1,18:PRINT
    STRING$(30," ");:GOTO 890             [2D24]
930 IF LEN(tibr$)>5 THEN LOCATE 1,18:PRI
    NT STRING$(30+LEN(tibr$)," "):GOTO 8
    90                                    [A5F4]
940 IF RIGHT$(tibr$,1)="." THEN LOCATE 1
    ,18:PRINT STRING$(30+LEN(tibr$)," ")
    ;:GOTO 890                            [0224]
950 tibr=VAL(tibr$):IF tibr=0 OR tibr>20
    0 OR tibr<10 THEN LOCATE 1,18:PRINT
    STRING$(35," "):GOTO 890              [8572]
960 mast=mast/10                          [A0EC]
970 vg=INT(vg*1000)                       [4EBC]
980 ho1=auho-mast*2                       [DBE2]
990 ho2=vg/ho1                            [7B38]
1000 ti1=tibr-mast*2                      [BD36]
1010 ti2=ho2/ti1                          [CEE8]
1020 br2=ti2+2*mast                       [4CE2]
1030 LOCATE 1,20:PRINT CHR$(24);"Tiefe o
     der Breite:"ROUND(br2,2);" cm";CHR$
     (24)                                 [4B48]
1040 LOCATE 40,14:PRINT CHR$(24);"Zuschn
     ittmasse...";CHR$(24)                [BA64]
1050 seiw1=br2:seiw2=auho-2*mast          [89BC]
1060 LOCATE 40,16:PRINT"2 Stueck ";:PRIN
     T USING"###.##";seiw1;:PRINT" cm *
     ";:PRINT USING"###.##";seiw2;:PRINT
     "cm"                                 [4366]
1070 depl1=tibr:depl2=br2                 [4400]
1080 LOCATE 40,18:PRINT"2 Stueck ";:PRIN
     T USING"###.##";depl1;:PRINT" cm *
     ";:PRINT USING"###.##";depl2;:PRINT
     "cm"                                 [CD22]
1090 frpl1=auho-2*mast:frpl2=tibr-2*mast
                                          [5886]
1100 LOCATE 40,20:PRINT"2 Stueck ";:PRIN
     T USING"###.##";frpl1;:PRINT" cm *
     ";:PRINT USING"###.##";frpl2;:PRINT
     "cm"                                 [CB42]
1110 LOCATE 30,24:PRINT"Neue Berechnung
     <1>"                                 [8368]
1120 LOCATE 30,25:PRINT"Zum Hauptmenue<2
     ><2>"                                [7042]
1130 awa$=INKEY$:IF awa$="" THEN 1130     [C63C]
1140 mast=mast*10:vg=vg/1000              [F072]
1150 IF awa$="2" THEN 200                 [8A4E]
1160 IF awa$<>"1" THEN IF awa$<>"2" THEN
      LOCATE 30,25:PRINT STRING$(30," ")
     :GOTO 1120                           [AB10]
1170 GOTO 470                             [97BA]
1180 '*****************************       [A3FA]
1190 '**        Zweiter  Teil       **    [2B24]
1200 '*****************************       [27EC]
1210 '**      Frequenzweichen       **    [508C]
1220 '*****************************       [B9F0]
1230 MODE 2                               [3BBA]
1240 GOSUB 5660                           [3DB0]
1250 LOCATE 45,4:PRINT CHR$(24);"<12>"    [EC4E]
```

»Boxenkit« hilft Ihnen beim Bau von Lautsprecherboxen

Ergänzen Sie jetzt Ihre ★HAPPY★ COMPUTER-Sammlung

Schaffen Sie sich ein interessantes Nachschlagewerk und gleichzeitig ein wertvolles Archiv!

Kennen Sie alle »Happy Computer«-Ausgaben von 1985? Suchen Sie einen ganz bestimmten Testbericht? Oder haben Sie einen Teil eines interessanten Kurses versäumt? Suchen Sie nach einer speziellen Anwendung?
Damit Sie jetzt fehlende Hefte mit »Ihrem« Artikel nachbestellen können, finden Sie auf diesen Seiten eine Zusammenstellung aller wesentlichen Artikel der noch lieferbaren Ausgaben. Und so kommen Sie schnell an die gewünschten Ausgaben: Prüfen Sie, welche Ausgabe in Ihrer Sammlung noch fehlt, oder welches Thema Sie interessiert. Tragen Sie die Nummer dieser Ausgabe und das Erscheinungsjahr (z.B. 2/85) auf dem Bestellabschnitt der hier eingehefteten Bestell-Zahlkarte ein. Die ausgefüllte Zahlkarte einfach heraustrennen und Rechnungsbetrag beim nächsten Postamt einzahlen. Ihre Bestellung wird nach Zahlungseingang umgehend zur Auslieferung gebracht.

Stichwort	Titel	Seite/Ausgabe
	Aktuelles	
Computer	Amiga — ein Traumcomputer wird Wirklichkeit	9/10
	Atari: Lage gefestigt	14/11
	Der »Plus/4« ist endlich da	12/2
	Grundstein einer neuen Linie und kein zweiter PC	13/10
	Konsequentes Chaos (Der deutsche QL)	14/10
DFÜ	Akustikkoppler für C 64	9/1
	Ascom-Koppler jetzt auch für Atari	20/8
	Ein Anschluß unter dieser Nummer (Mailbox Nummern)	159/3
	Mailboxbetrieb in den USA	22/10
	Neues DFÜ-Programm für den Spectrum	22/10
	Nullmodem zum Aufstecken	12/1
Software	Atari-Schreiber jetzt für 520 ST	14/12
	Software fast zum Nulltarif	10/1
	Träume werden wahr (Schneider-Neuheiten aus England)	9/12
Drucker	Mac Inker, der sparsame Drucker	12/12
Floppy	Commodore-Floppy auf Trab gebracht	9/1
	Diskettenlaufwerk für den Sharp MZ-800	12/1
	Quick Disk — Die Floppy-Alternative (MSX)	20/4
Erweiterung	Mini-Expansion-Box für TI 99/4A	11/1
MSX	Das Musikwunder (Yamaha CX-5)	141/2
	Der Billig-MSX von Philips kommt	50/1
	CP/M mit MSX-Computer: so geht's	141/2
	Ein komplettes System von Philips	19/8
	Flotter Dreier (Sanyo, Goldstar und Canon)	23/5
	MSX-Mix	45/3
	Mit dem fliegenden Teppich auf Erfolgskurs	15/10
Bücher	Bücher zur DFÜ	111/3
	Bücher zum Denken (KI)	120/10
	Messeberichte	
	Die neuesten Heimcomputer (Winter-CES)	9/3
	Funkausstellung in Berlin: MSX war Trumpf	9/11
	Kampf der Kolosse (Winter-CES — Teil 1)	9/4
	Sommer-CES 1985: Weiche Welle in Chicago — Teil 1	9/8
	Software-Jackpot (Winter CES — Teil 2)	9/5
	Software-Super-Show in London (PCW-Show)	12/11
KI	Künstliche Intelligenz in Wiesbaden (AI Europa)	13/12
Musik	Musikmesse Frankfurt: Midi marschiert	22/5
	Interviews	
	David Crane (Ghostbusters Autor)	17/5
	Interview mit den »Print Shop«-Machern	14/8
	Jack Tramiel (Chairman Atari)	11/2
	Hardware-Tests	
Drucker	Bewußt robust (Europrint K 6311 FT)	31/5
	Software-Tests	
Textverarb.	Ein Textprogramm, das sich lohnt (Homeword/C 64)	77/4
	Drei Drucker im Test (STX 80, Gemini 10X, CP-80X)	16/1
	(Nachhall auf Seite 149 in 4/85) DWX 305: Schönschrift zum Niedrigpreis	18/2
	Eine heiße Verbindung (EP 22, EP 44, EXD 10)	26/5
	Kompakt und leise: Matrixdrucker GLP (Centronics)	24/1
	Regenbogenfarben — wie gedruckt (Okimate 20)	154/10
	Schön oder schnell (Horizon HX 80)	21/3
	Spectrum mit starken Typen (Gabriele 9009)	126/11
	Zwei Drucker für den Schneider (NLQ 401, GP 500 CPC)	112/8
Computer	Chinese mit britischem Paß (Triton 64)	22/2
	Der Musik Maestro (Yamaha CX-5)	28/4
	Der Neue: Commodore PC 128	46/5
	Der »neue« Spectrum	31/1
	Ein »Einsteiger« aus Taiwan (BIT-90)	16/2
	Joyce — Schneiders Einstieg in die Welt der PCs	24/11
	Quantensprung im Schneckentempo (QL dt. Version)	180/11
	Koreaner mit Deutsch-Talent (Ce-Tec/MSX)	18/3
	Schneiders neue Dimension (CPC 6128)	24/10
	Sharps Jüngster (Sharp MZ-800)	20/1
	Spectrum plus oder Spectrum minus	24/4
	Viel Computer für wenig Geld (Schneider CPC 664)	113/8
	YC-64: Fernöstlicher Biedermann (MSX Computer)	20/2
	Wer ist wer? (Atari 520 ST+ und 260 ST)	16/12
	Wie musikalisch ist mein Heimcomputer?	148/11
Laufwerke	3-Zoll-Erfahrungen (MCD-1-Floppy für Spectrum) (Discovery/Spectrum)	22/1
	Ein ungleiches Paar (Spectrum — VIC 1541 Interface)	21/4
	Lauf, Floppy, lauf! (SpeedDos plus/C64)	45/12
	Preiswertes Spectr um Floppysystem (Viscount System)	21/2
	Spectrum Diskettensystem im Plus-Look	20/3
	VC 1541 wird zur Rennfloppy	42/4
Recorder	Der Spectrum Sprinter (Datenrecorder: Sprint)	28/1
	Ein billiger Speicher für alle (Recorder MC 3810)	30/5
DFÜ	DFÜ auch mit dem TI (RS 232 für TI 99/4A)	25/5
	Kommunikation mit dem Spectrum	32/4
	Spartanisch aber gut (Ascom Akustikkoppler)	158/3
Sonstiges	Computer steuert Modelleisenbahn	176/11
	Der andere Weg (Spectrum Tastatur)	19/3
	Faszination der Technik (Fischer Technik Roboter)	44/11
	Famose Formel für den C64 (Formel 64)	40/12
	Grafpad Supergrafik für den Spectrum	16/3
	Haltet den Dieb (Alarmanlage für C 64, VC 20)	29/1
	Ohren oder Tasten? (Voice Command Modul/C 64)	40/10
	Peripherie für MSX (Plotter, 3½-Zoll-Floppy) (Joysticks im Vergleichstest)	26/1
	Roboter, Technologie der Zukunft (Fischertechnik)	45/4
	Starker Arm für Heimcomputer (Teach Robot)	38/4
	Tafelfreuden für Grafik-Gourmets (Atari Maltafel)	14/1/
	Vom Piepmatz zum Mini-Orchester (Spectrum Sound)	15/2
	Software-Tests	
Textverarb.	Ein Textprogramm, das sich lohnt (Homeword/C 64)	77/4
	Jedem seine Zeitung (The Newsroom)	118/8
	Schreiben ohne Frust	46/2
	Textverarbeitung für jedermann (Homewriter für MSX)	137/1
Sprachen	Basic-Erweiterung zum Spartarif (Aztec Basic/C 64)	76/4
	Drei Assembler für Atari-Computer im Vergleich	30/3
	Fortschritt rückwärts (CP/M-80 Emulator für 520 ST)	138/11
	Hisoft-Pascal jetzt Microdrive-kompatibel (Spectrum)	56/2
	Logo für den Atari 520 ST	134/11
	Mallard-80-Basic — ein starkes Stück	28/11
	Maschinensprache ist keine Zauberei (CPC 464)	107/8
	Prozessor-Welt von morgen: C 64 simuliert 68000	42/10
	Spezielles Spiele-Basic für den Spectrum	143/5
	Welches Basic für meinen MZ-700?	48/2
	Zwölf Farben in Mode 2 (Color Star für CPC 464)	110/8
Utilities	Das Programm, das Programme macht (Progressor)	33/5
	Disketten-Doktor für den C 128	42/12
	Quicksave für Spectrum	137/4
	SM-Kit — Das Werkzeug für Lehrling und Meister (C 64)	138/1
	Software-Knackern dazwischengepfuscht (Apple II)	27/3
Grafik	Beeindruckend (Print Shop — Druckprogramm)	50/2
	Die Maus bringt Farbe auf den Bildschirm (Apple)	52/2
	Grafik grandios (Malprogramm Blazing Paddles)	28/3

Stichwort	Titel	Seite/Ausgabe
	Koala Bilder zum Anfassen (Hardcopy-Programm)	57/2
	Mit dem Joystick programmiert (Designers Pencil)	140/5
	Viel Grafik für wenig Geld (Graphics Basic und Supergrafik 64 für C 64 im Vergleich)	44/2
	Vorsicht Kamera! (Take 1, Trickfilm Designer)	136/8
DFÜ	Apple II sucht Anschluß	154/3
	Contact 64 — Die Software zum Ascom-Koppler	142/5
	Spectrum auf Draht (DFÜ Vergleichstest)	124/8
Astronomie	Spectrums Sternstunden	34/3
	Sterngucker	158/10
Schach	Schachmatt per Telefon	156/10
	Spiele-Tests	
	Amazon	145/5
	Archon II: Adept	126/2
	Asylum	144/3
	Athletic Land	146/1
	A View to a Kill	169/10
	Ballblazer	167/10
	Boulder Dash	125/2
	Bounty Bob strikes back	139/8
	Cavelord	124/2
	Crazy Train	144/1
	D-Bug	118/2
	Deus ex Machina	146/4
	Don't buy this	168/12
	Doomdark's Revenche	148/5
	Dorodon	142/3
	Dragonsden	124/2
	Elektro Freddy	145/1
	Elite	164/10
	Eureka	144/4
	Fahrenheit 451	148/8
	Five-a-Side Football	166/10
	Formula One	140/8
	Frank Brunos Boxing	166/10
	Frankie goes to Hollywood	162/10
	Fruity Frank	145/4
	Ghettoblaster	169/11
	Ghostbusters	138/3
	Ghost Chaser	170/11
	Great American Cross Country Road Race	168/11
	Hacker	167/12
	H.E.R.O.	149/5
	Hyper Sports 1	143/3
	Karateka	146/4
	Kennedy Approach	168/12
	Knight Lore	143/3
	Macbeth	144/4
	Mask of the Sun	122/2
	Match Day	150/5
	Mindshadow	141/8
	Mr. Do	167/10
	Monster Trivia	168/10
	Nick Faldo plays the Open	169/11
	Nightshade	169/12
	Nodes of Yesod	169/12
	On Court Tennis	150/5
	Pitfall II	148/5
	Rama	145/5
	Rescue on Fractalus	168/10
	Rocket Ball	140/8
	Rockford's Riot (Bolder Dash II)	168/11
	Rock'n Bolt	139/8
	Sherlock Homes	121/2
	Seastalker	147/1
	Serpent's Star	142/4
	Software Star	165/11
	Spelunker	142/3
	Standing Stones	145/4
	Summer Games II	133/8
	Super Pipeline II	141/8
	The Ancient Art of War	149/5
	The Fourth Protocol	165/11
	The Hitchhiker's Guide to the Galaxy	138/4
	The Little Computer People Projekt	170/12
	The Way of exploding Fist	169/10
	Tour de France	170/11
	Where in the World is Carmen San Diego	163/11
	Whistler's Brother	141/3
	White Lightning	148/1
	Winter Games	164/12
	World Championship Boxing	170/12
	ZimSalaBim	141/3
	Spiele Tips	
	Abenteuer im Weltraum	152/8
	Amazon	172/10
	Aztec Challenge	147/4
	Aztec Tomb	147/4
	Aztec Tomb	173/10
	Beach Head	85/1
	Death in the Caribbean	142/8
	Death in the Caribbean	172/12
	Doomdark's Revenge	142/8
	Everyone's a Wally	173/10
	Ghostbusters	140/3
	Ghostbusters	147/4
	Ghostbusters	152/5
	Hampstead	172/12
	Heros of Karn	173/12
	Hexenküche	173/12
	Hobbit	146/3
	Hulk	143/8
	Hunch Back	85/1
	Karateka	172/12
	Lode Runner	174/11
	Lode Runner	174/12
	Mask of the Sun	173/11
	Masquerade	144/8
	Mindshadow	174/11
	Miner 2049er	147/4
	Pirate Adventure	126/2
	Pitfall	147/4
	Pitfall II	144/8
	Pitfall II	174/10
	Sabre Wulf	85/1
	Sands of Egypt	173/11
	Sands of Egypt	174/12
	Schloß des Grauens	152/5
	Secret Mission	173/11
	Ship of Doom	172/12
	Spelunker	144/8
	Strip Poker	152/5

Stichwort	Titel	Seite/Ausgabe
	Summer Games	[illegible]
	Summer Games	[illegible]
	Super Huey	[illegible]
	The Dallas Quest	[illegible]
	The Institute	[illegible]
	The Institute	[illegible]
	The Quest	[illegible]
	The Witness	[illegible]
	Time Maschine	[illegible]
	Ultima II	[illegible]
	Ultima II	[illegible]
	Ultima III	[illegible]
	Valhalla	[illegible]
	Whistler's Brother	[illegible]
	Zeppelin	[illegible]
	ZimSalaBim	[illegible]
	Zork	[illegible]
	Listings	
Anwendung	Alle Neune (Jahresauswertung-Kegeln/C 64)	[illegible]
	(Adreßverwaltung/C 64) Besseres Basic ganz einfach Software Basic 3.0/C 64)	[illegible]
	Datenbank mit freiem Zugriff (C 64)	[illegible]
	Der Halleysche Komet kommt (MSX)	[illegible]
	Die Mini-Textverarbeitung (Spectrum)	[illegible]
	Nachhall auf Seite 160 in 9/85	
	Do-it-yourself-Datenverwaltung (Mainfile II/C 64)	[illegible]
	Einblick ins Innenleben (Disassembler/CPC 464)	[illegible]
	Eine tolle Textverarbeitung für den Schneider (464)	[illegible]
	Nachhall auf Seite 85 in 5/85	
	Funktionen optisch aufbereitet (VZ-200/Laser)	[illegible]
	Geregelte Finanzen mit dem Commodore 64	[illegible]
	Nachhall auf Seite 117 in 8/85	[illegible]
	Morse-Decoder für Funkamateure (Spectrum)	[illegible]
	Nebenkostenabrechnung (C 64)	[illegible]
	Nachhall auf Seite 80 in 12/85 Optik mit Simons Basic (C 64)	[illegible]
	Programme in Reih' und Glied (C 64)	[illegible]
	Suchen, nein danke (Dateiverwaltung/CPC 464)	[illegible]
	Nachhall auf Seite 117 in 8/85	
	Transistor-Schaltungen berechnen (L.d.M./Spectrum)	[illegible]
	Turbo-Basic-Interpreter für Atari 800XL (L.d.M.)	[illegible]
Grafik	Apple IIc-HiRes-Grafik auf dem Drucker	[illegible]
	Bewegte Grafik mit drei Befehlen (CPC 464)	[illegible]
	Farbspielereien (Atari)	[illegible]
	Grafikentzerrung für Matrixdrucker (Spectrum)	[illegible]
	Grafik-Window bekommt Nachwuchs (C 64)	[illegible]
	Nachhall auf Seite 80 in 12/85	
	Grafikzauber (Apple II)	[illegible]
	Hires Fantasy (C 64)	[illegible]
	Rosetten-Grafik für den Spectrum	[illegible]
	Schnelle Grafik aus dem Compiler (L.d.M./C 64)	[illegible]
	Schöne schnelle Grafik (Grafik-Paket/C 64)	[illegible]
	Solar-Painter (Spectrum)	[illegible]
	Sprite-Editor (C 64)	[illegible]
	Zauber der Farben mit Magic Painter (L.d.M./Atari)	[illegible]
	Nachhall auf Seite 85 in 5/85	
	Zaubereien auf dem Bildschirm (L.d.M./Grafik/CPC 464)	[illegible]
	Zeichenroutine für Kreise und Ellipse (CPC 464)	[illegible]
	Zykloide für Grafiker und Mathematiker (C 64)	[illegible]
	Nachhall auf Seite 79 in 12/85	
Spiel	Das Haus des Magiers (C 64)	[illegible]
	Dasher, der Volltreffer (L.d.M./C 64)	[illegible]
	Nachhall auf Seite 117 in 8/85	
	Der rasende Raider (C 64)	[illegible]
	Diamantenfieber (L.d.M./Atari 48 KByte)	[illegible]
	Nachhall auf Seite 85 in 5/85	
	Die Abenteuer eines rasenden Reporters (Report/C 64)	[illegible]
	Geröllheimer (Atari)	[illegible]
	Gespensterjagd im Schneider (CPC 464)	[illegible]
	Nachhall auf Seite 85 in 5/85	
	Lumberjack Larrys Abenteuer in Bagdad (S.d.M/C 64)	[illegible]
	Kalte Zeiten (Wintry Screen/C 64)	[illegible]
	Kneipe zum hastigen Kellner (VC 20)	[illegible]
	Mit dem Apple auf die Trainerbank (Aktion Apfelsaft)	[illegible]
	Mit dem Atari-Computer auf Ölsuche (Atari)	[illegible]
	Mücke mit Tücke (C 64)	[illegible]
	Nachtflug (Spectrum)	[illegible]
	Nachhall auf Seite 85 in 5/85 Niemandsland (C 64)	[illegible]
	Pokerface für 16 KByte (Spectrum)	[illegible]
	Psycho — die Macht des Geistes (C 64)	[illegible]
	Nachhall auf Seite 80 in 12/85	
	Rennfahrer mit dem Joystick (Driver/C 64)	[illegible]
	Rettet den letzten Baum (Insekt defense/C 64)	[illegible]
	SAM — der Mann von der Baustelle (L.d.M./CPC 464)	[illegible]
	Nachhall auf Seite 79 in 12/85	
	Schatzhöhle (Atari 800XL)	[illegible]
	Vorsicht Hochwasser (Aquantor/L.d.M./C 64)	[illegible]
	Über den Wolken (Flugplanung/C 64)	[illegible]
	Wortsuchspiel (Spectrum)	[illegible]
Tips & Tricks	AMPEL — grünes Licht für Atari-Maschinen-Programme	[illegible]
	Auf Trap gebracht (CPC 464)	[illegible]
	Basic bequem (C 64)	[illegible]
	Basic-Compactor (Spectrum)	[illegible]
	Nachhall auf Seite 80 in 12/85	
	Basic-Plus: Applesoft-Basic-Erweiterung (Apple II)	[illegible]
	Beim C 64 piepst es	[illegible]
	Bilder richtig konservieren (CPC 464)	[illegible]
	Bildschirmtrick für den Commodore 64	[illegible]
	Byte-Shifter (Spectrum)	[illegible]
	Dateien hin- und hergerissen (Atari-IBM)	[illegible]
	Der neue Checksummer ist da (C 64)	[illegible]
	Der neue Checksummer ist da (C 64)	[illegible]
	Der neue Checksummer (C 64)	[illegible]
	Deutsche Sonderzeichen unter CP/M (CPC 464)	[illegible]
	Die Maltafel wird zur Maus (Atari)	[illegible]
	Nachhall auf Seite 49 in 11/85	
	Disk-Help für die schnelle Hilfe (Atari)	[illegible]
	Disk- und DOS-Utility für alle Atari-Computer	[illegible]
	Drei Tricks für MSX	[illegible]
	Ein langes Gesicht für den C 64 (Longscreen 64)	[illegible]
	Fehlerhilfe mit HELP & TRACE (VC 20)	[illegible]
	Fensterlkünstler (C 64)	[illegible]
	Fettschrift für den 48 KByte-Spectrum	[illegible]
	Find Label (Spectrum)	[illegible]
	Flotte Primzahlen in Hisoft Pascal (Spectrum)	[illegible]
	Fußball-Manager für Commodore 64	[illegible]
	Grafik-Hardcopy in vierfacher Größe (C 64)	[illegible]
	Kostenlose Speichererweiterung (C 64)	[illegible]
	Listen leicht gemacht (C 64)	[illegible]
	Make DATA für den Spectrum	[illegible]
	Maschinencode-Routinen in Basic umgesetzt (CPC 464)	[illegible]
	Microdrive-Aufwertung (Spectrum)	[illegible]
	Nachhall auf Seite 79 in 12/85	
	Mondlandung (C 64)	[illegible]
	Musik und Farbe (C 64)	[illegible]
	Nie mehr Listingkummer mit dem Checksummer (C 64)	[illegible]
	Nie mehr Listingkummer mit dem Checksummer (C 64)	[illegible]
	Prima Werkzeug für den Programmierer (C 64)	[illegible]
	Programmtransfer leicht gemacht (CPC 464)	[illegible]
	Proportionalschrift für den Spectrum	[illegible]
	Protokoll auf dem Drucker (CPC 464)	[illegible]
	RAM-Disk für Atari 800XL	[illegible]
	Ran an den Userport (C 64)	[illegible]
	Renumber 64 (C 64)	[illegible]
	Nachhall auf Seite 117 in 8/85	
	Rock me Amadeus (C 64)	[illegible]
	RSX-Befehle ohne »@« (Schneider)	[illegible]
	Schilderwald (Plakatschrift/C 64)	[illegible]
	Schluß mit der Eintönigkeit (C 64)	[illegible]
	Nachhall auf Seite 80 in 12/85	
	Spectrums COPY besser nutzen	[illegible]
	Spectrumtasten mit Funktionen belegt (Spectrum)	[illegible]
	Spectrum Tips & Tricks	[illegible]
	Sprachkurs für Commodore-Basic (C 64)	[illegible]
	Super-Merge für Commodore 64	[illegible]
	Nachhall auf Seite 160 in 9/85	
	Super-Saver (C 64)	[illegible]
	Statuszeile mit Uhr (Atari)	[illegible]
	Tasword 464 mit DIN-Tastatur (CPC 464)	[illegible]
	Texte auch im Grafikmodus (Atari)	[illegible]
	Tippen mit dem Plotter (C 64)	[illegible]
	Tips & Tricks rund um den Schneider	[illegible]
	Töne aus dem Atari	[illegible]
	Variablendump für Atari (Atari)	[illegible]
	Variablen-Transfer (Spectrum)	[illegible]
	Verflixter Listschutz (C 64)	[illegible]
	Nachhall auf Seite 80 in 12/85	
	Vom Maschinencode zum Basic-Programm (C 64)	[illegible]
	Wie die Bilder laufen lernten (Atari)	[illegible]
	Zeilenakrobatik auf dem Schneider	[illegible]
	Zwei SCREEN$ im schnellen Wechsel (Spectrum)	[illegible]
	ZX81-Utility: Nützliches für Aufsteiger (C 64)	[illegible]
	30 tolle Maschinencode-Routinen (Spectrum)	[illegible]
	Grundlagen	
Speicher	Daten am laufenden Band	[illegible]
	Daten auf der schnellen Scheibe	[illegible]
	Floppy gegen Kassette	[illegible]
	Selbst geschraubt ist halb gespart	[illegible]
	So arbeitet das 1050-Laufwerk von Atari	[illegible]
	So liest und schreibt die 1541	[illegible]
	Speichermedium Endlosband	[illegible]
	Tips, Tricks und Todsünden	[illegible]
	Wohin in Zukunft mit Bits und Bytes	[illegible]
Monitore	Farbmonitore — buntes Fenster zum Computer	[illegible]
	Monitore: Richtig geplant, gekauft und genossen	[illegible]
Drucker	Blitzsaubere Schrift mit Laserlicht (Laserdrucker)	[illegible]
	Die »heißen« Drucker (Thermodrucker)	[illegible]
	Farbspiele für Farbdrucker	[illegible]
	Mit leisen Tönen (Tintenstrahldrucker)	[illegible]
	Scharfe Nadel, spitze Typen (Matrix und Typenrad)	[illegible]

Titel	Seite/Ausgabe
Auf einen Blick: Logo-Befehle	132/2
Befehlserweiterung für RSX (CPC 464)	34/10
CP/M — Ein Betriebssystem	84/8
Fenster in die Zukunft: Basic auf dem 520 ST	132/12
Logo-Spielerei oder ernsthafte Alternative	110/1
RSX — Maschinensprache mit Komfort	34/11
Begriffe aus der DFÜ	151/3
Datenübertragung im schnellen Gleichschritt	144/11
Beethoven — Bit für Bit	152/11
Der Weg zum Kabelorchester	157/11
Das Interface 1 ROM und seine Nutzung	158/4
Der Commodore 64 kann einfach alles	59/4
Der 18er und sein RAM	43/12
Ein großes Abenteuer: Das Adventure	128/2
Messen + Steuern = Regeln	146/11
Schnittstellen — was sind das eigentlich	36/4
So bauen die Spiele-Baukästen	32/5
Vom Traum zum Heimcomputer (68000 Prozessor)	20/11
Weiche Hardcopy (Schneider)	74/12
Welcher Computer spielt am besten?	158/12
1, 2, 3 — Kalkulieren mit der Hand ist nun vorbei	80/8
Allgemeine Themen	
Der Computer — Ein moderner Trichter?	116/2
Schule mit Computer	118/10
Keine Angst vor DFÜ	153/3
Amiga Spiele Premiere	161/12
Bits auf Abwegen	147/11
Computer als Briefträger	148/3
Der C 64 im C 128	51/11
Ein teures Vergnügen (DFÜ-Kosten)	154/3
Happy-Sportspielführer	137/8
Heimcomputer aus zweiter Hand	142/12
Mehr als ein Computer (Die Commodore Story)	49/4
Raupkopierer gegen den Rest der Welt	126/10
Software (fast) geschenkt	151/10
Software-Piraterie	23/8
Software-Volltreffer	144/12
Software zum Spartarif	153/10
Spiele auf der schwarzen Liste	160/11
Vom Heimcomputer-Freak zum EDV-Spezialisten	35/2
Vom Hobby zum Geldregen	39/2
Vom Abenteuer, ein Abenteuer zu schreiben	42/2
Wenn mal was schiefgeht	140/12
Wissenswertes, Fragen und Antworten zum 128er	52/11
Zubehör und Software — das »kleine« Geschenk	39/1/
Zu viel Kontrolle	150/3/
1985 — Das Jahr der Eisenbahn	154/4
Kurse	
Teil 1: Der Einstieg für Einsteiger	40/3
Teil 2: Die Schildkröte lernt laufen	151/4
Teil 3: Die Schildkröte wird erwachsen	153/5
Pascal für Schüler und Lehrer	86/8
Pascal für kluge Köpfe/Teil 2	121/10
Pascal für kluge Köpfe/Teil 3	124/11
Schnelle Grafik für Atari Computer	124/10
Musik mit Poke und Peek/Teil 1	54/3
Musik mit Poke und Peek/Teil 2	53/4
Musik mit Poke und Peek/Teil 3	56/5
Lernen Sie Ihren Commodore 64 kennen/Teil 1	59/5
Lernen Sie Ihren Commodore 64 kennen/Teil 4	45/8
Lernen Sie Ihren Commodore 64 kennen/Teil 6	45/10
Lernen Sie Ihren Commodore 64 kennen/Teil 7	56/11
Ohne Fleiß kein Kreis/Teil 1	48/12
Kein Buch mit sieben Siegeln/Teil 1	156/5
Kein Buch mit sieben Siegeln/Teil 4	105/8
Zugüberwachung per Computer/Teil 1	155/4
Zugüberwachung per Computer/Teil 2	51/5
Basteln	
Atari 520 ST auf Abwegen	23/12
Bilder aus dem Weltall (Schneider)	32/12
Dem User Port geht ein Licht auf (C 64)	54/11
Fehler in der Spectrum Hardware	43/8
Gute Verbindung mit dem Schneider (PIO-Interface)	28/10
Lightshow mit dem Commodore 64	44/5
Multitalent für den Joystickanschluß (Spectrum)	30/2
Nachhall auf Seite 85 in 5/85	
Nachhall auf Seite 77 in 7/85	
Neue Geräteadresse für das 1541 Laufwerk (C 64)	62/10
Nie wieder Angst (Alarmanlage C 64)	48/3
Nachhall auf Seite 80 in 12/85	
Schalten und walten mit dem Atari (Schaltinterface)	114/10
Schreiben mit Schreibmaschinenqualität (C 64)	26/2
Schreibschutz-Schalter (Atari 810 Floppy)	24/3
Schreibschutz-Schalter (Atari 1050 Floppy)	107/11
Sieben auf einen Port (7 Segment Anzeige/Spectrum)	24/2
Sparen am richtigen »Drucker-Ende« (Sinclair)	23/3
Verbesserte Cursorsteuerung beim Spectrum	29/2
Zwei Joysticks für ein Halleluja (CPC 464)	31/5
Marktübersichten	
Erweiterungen zum TI 99/4A	40/1
Marktübersicht Atari	46/1
Rund um den Atari	128/11
Jede Menge Software	132/11
Anschluß gesucht: Peripherie für ZX81 und Spectrum	48/1
Interfaces für den Commodore 64	49/1
Der Computer mit dem großen Zubehör	56/4
Akustikkoppler, preiswert wie noch nie	160/3
Druckerparade	129/10
Nachhall auf Seite 80 in 12/85	
Marktübersicht Monitore	136/5
Nachhall auf Seite 80 in 12/85	
Musiksoftware	151/11
Softladen (Die neusten Programme und ihre Preise)	32/1
So viel Software (Heimsoftware für Heimcomputer)	150/12
Spiele aus dem Baukasten (Construction Sets)	38/5
Welcher Computer zum Weihnachtsfest?	136/12
Wettbewerbe	
Aktion Apfelsaft	29/1
Bildergalerie	106/1
Bildergalerie (Nachlese)	142/2
Bithoven-Festival	46/3
Bithoven-Festival	128/8
Der Computer als Steuermann	48/11
Der schönste Titel von 1984	108/1
Der schönste Titel von 1984	135/5
Diskettenwettbewerb	176/10
Happy Computer Leserwettbewerb	130/8
Happy Computer Leserwettbewerb	20/12
Ihr Einsatz (Die beste Anwendung)	104/1
Leserumfrage — Taschenrechner	70/10
Probleme auf der Wöraalm	179/11
Spiel des Jahres	148/4
Steno mit dem Computer	41/5
Was steuern, wie regeln?	46/11
Wer gewinnt den goldenen Besenstiel	172/11
Leserforum	
Atari-Tips	102/1
Autostart für VC 20	103/1
Basicode-2 für MZ-700	77/2
Basic-Speicher ohne Boden (C 64)	185/11
Basic und HiRes-Grafik (C 64)	160/12
Commodore-Ecke	117/10
Eingabezeile beim Spectrum speichern	110/3
Gedächtnislücke beim ZX 81	35/4
gtext 64 an RX 80 angepaßt	103/1
Joystickprobleme beim VC 20	77/2
LPRINT III — Fehlerloses Drucken auch ohne EPROM	159/12
Probleme mit den langen Zeilen (C 64)	185/11
Probleme mit 800XL	159/12
Sprite-Kollision (C 64)	160/12
Stereo aus dem Commodore 64	110/3
Tip für Oric 1	103/1
Unvollständige Adresse beim ZX 81	77/2
VC 20 und Videokamera am Monitor	103/1

Die Ausgaben 6/85, 7/85 und 9/85 sind bereits vergriffen und nicht mehr lieferbar!

Auch die bisher erschienenen Sonderhefte können Sie jetzt direkt bestellen:

SONDERHEFT 01/84: SINCLAIR
Unentbehrliche Informationen zu den Sinclair Computern ZX81 und Spectrum.

SONDERHEFT 01/85: SPECTRUM
Anwendungsbezogene Listings und Tips & Tricks für alle Spectrum-Fans.

SONDERHEFT 02/85: SCHNEIDER 1
Eine Fülle wertvoller Beiträge und Listings für alle Schneider-Anwender.

SONDERHEFT 03/85: SPIELE
Ein Super-Nachschlagewerk für alle Spiele-Fans mit 100 Spielen im Test und großer Marktübersicht.

SONDERHEFT 01/86: SCHNEIDER 2
Noch mehr Tips und Tricks für Einsteiger und Fortgeschrittene mit vielen interessanten Programm-Listings.

SONDERHEFT 02/86: ATARI 1
Besonders 800 XL- und 130 XE-Fans erwarten jede Menge Anwendungs- und Spiele-Listings sowie Informationen.

SONDERHEFT 03/86: 68000er
Umfassende Informationen zur neuen Computer-Generation und eine große Vergleichstabelle, die im Detail über alle 68000er informiert.

SONDERHEFT 04/86: SCHNEIDER 3
Eine Erweiterung für alle Schneider-Anwender, Super-Programm-Listings und großer Einsteiger-Teil.

SONDERHEFT 05/86: PROGRAMMIERSPRACHEN
Fuß fassen in »Pascal«, »C« und »Forth« mit jeweils einem grundlegendem Kurs und vielen Anwendungs-Listings.

SONDERHEFT 06/86: 68000er 2
Umfangreicher Listingteil, viele Informationen, Tips und Tricks für Anwender der 68000er-Computer.

SONDERHEFT 07/86: SCHNEIDER 4
Mit den Schwerpunkten Joyce und CP/M plus, Ratschlägen zur Vortex-Karte und vielen Tips & Tricks.

SONDERHEFT 08: COMPUTER ALS HOBBY
Wissenswertes für Einsteiger und zusätzliche Informationen zur Fernsehsendung Computerzeit.

SONDERHEFT 09: 68000er 3
Mit den Schwerpunkten Sound- und Videodigitalisierung und Spieleprogrammierung.

SONDERHEFT 10/86: SCHNEIDER 5
Der neue Schneider-PC wird vorgestellt. Wieder viele Hilfestellungen und Kurse.

SONDERHEFT 11/86: SPIELE-TESTS
Alles über aktuelle Spieletests, Computerprogramme, Grafik- und Musik-Software.

SONDERHEFT 12/86: 68000er 4
Ausführliche Testreihe aller Grafikprogramme für Atari ST, Amiga und Sinclair QL sowie viele Grundlageninformationen zu diesen Computern.

SONDERHEFT 13: SCHNEIDER 6
Diskettengrundlagen-Kurs. Reiche Auswahl an Programmen für CPC. Schneider CPC oder PC: Fakten und Vorteile.

SONDERHEFT 14: SOFTWARE
Der Softwareführer 1987 für Ihre optimale Programmauswahl.

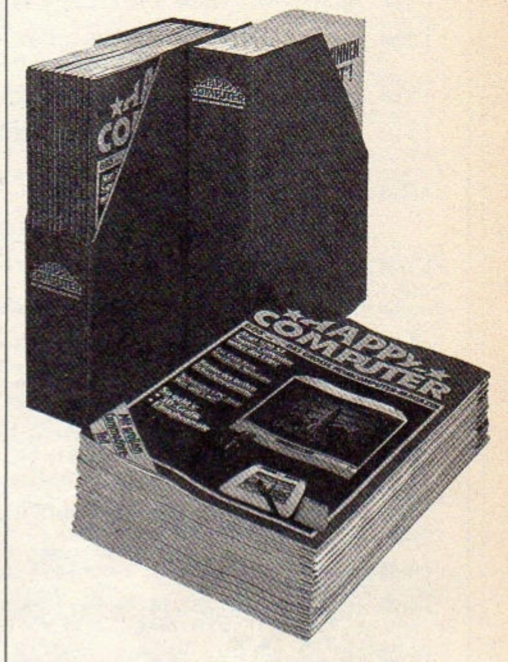
SONDERHEFT 15: HARDWARE-TESTHEFT
Über hundert Geräte für optimale Hardware-Auswahl im Test.

SONDERHEFT 16: SCHNEIDER 7
Das Super-Programm CPC-Giga-Cad: Dreidimensionales Zeichnen plus animierte Grafik.

SONDERHEFT 17: SPIELE-TESTS
Ausgewählte Spieleneuerscheinungen vorgestellt und kritisch unter die Lupe genommen.

Tragen Sie die Nummer des gewünschten Sonderheftes (z.B. 08/85) auf dem Bestellabschnitt der hier eingehefteten Bestell-Zahlkarte ein.

Am besten gleich mitbestellen: Die Happy-Computer-Sammelboxen



Für alle Leser, die »Happy Computer« regelmäßig kaufen, sammeln oder im Abonnement beziehen, gibt es ein interessantes Service-Angebot: die Happy-Computer-Sammelbox!

Mit dieser Sammelbox bringen Sie nicht nur Ordnung in Ihre wertvollen Hefte, sondern schaffen sich gleichzeitig ein interessantes und attraktives Nachschlagewerk. Ein kompletter Jahrgang (12 Ausgaben) paßt in eine der praktischen Sammelboxen!

Übrigens: Die Sammelbox ist nicht nur ein praktisches Aufbewahrungsmittel: Sie eignet sich auch hervorragend als Geschenk für Freunde und Bekannte zu vielen Anlässen.

```
1260 LOCATE 45,5:PRINT" WEICHEN... "        [A9C6]
1270 LOCATE 45,6:PRINT"<12>";CHR$(24)       [E516]
1280 LOCATE 45,9:PRINT"Zweiweg- oder Dre
     iwegweiche ?<2><2/3>"                  [54FC]
1290 a$=INKEY$                              [EDA4]
1300 IF a$<"2" OR a$>"3" THEN 1290          [E640]
1310 IF a$="3" THEN fff=3 :GOTO 1780 ELS
     E fff=2                                [E5D6]
1320 LOCATE 45,9:PRINT STRING$(36," "):L
     OCATE 45,9:PRINT"Z W E I W E G W E
     I C H E . . ."                         [ABC8]
1330 LOCATE 1,5:INPUT"Impedanz des Tieft
     oeners :",rt$                          [C056]
1340 IF LEN(rt$)>3 OR rt$=""THEN LOCATE
     1,5:PRINT STRING$(30+LEN(rt$)," ");
     :GOTO 1330                             [030E]
1350 IF ASC(LEFT$(rt$,1))<48 OR ASC(LEFT
     $(rt$,1))>57 THEN LOCATE 1,5:PRINT
     STRING$(30+LEN(rt$)," ");:GOTO 1330
                                            [1B44]
1360 IF LEN(rt$)=3 THEN x$=LEFT$(rt$,2):
     x$=RIGHT$(x$,1):IF x$<>"." THEN LOC
     ATE 1,5:PRINT STRING$(30+LEN(rt$),"
      "):GOTO 1330                          [AF82]
1370 IF rt$="0" THEN LOCATE 1,5:PRINT ST
     RING$(30," ");:GOTO 1330               [1AD2]
1380 IF LEN(rt$)>=4 THEN LOCATE 1,5:PRIN
     T STRING$(30+LEN(rt$)," "):GOTO 133
     0                                      [E884]
1390 IF RIGHT$(rt$,1)="." THEN LOCATE 1,
     5:PRINT STRING$(30+LEN(rt$)," ");:G
     OTO 1330                               [403C]
1400 rt=VAL(rt$)                            [B88C]
1410 IF rt<2 OR rt>32 THEN LOCATE 1,5:PR
     INT STRING$(30+LEN(rt$)," ");:GOTO
     1330                                   [EB98]
1420 LOCATE 1,7:INPUT"Impedanz des Hocht
     oeners :",rh$                          [3936]
1430 IF LEN(rh$)>3 OR rh$=""THEN LOCATE
     1,7:PRINT STRING$(30+LEN(rh$)," ");
     :GOTO 1420                             [62CA]
1440 IF ASC(LEFT$(rh$,1))<48 OR ASC(LEFT
     $(rh$,1))>57 THEN LOCATE 1,7:PRINT
     STRING$(30+LEN(rh$)," ");:GOTO 1420
                                            [1B00]
1450 IF LEN(rh$)=3 THEN x$=LEFT$(rh$,2):
     x$=RIGHT$(x$,1):IF x$<>"." THEN LOC
     ATE 1,7:PRINT STRING$(30+LEN(rh$),"
      "):GOTO 1420                          [6D3E]
1460 IF rh$="0" THEN LOCATE 1,7:PRINT ST
     RING$(30," ");:GOTO 1420               [0EBE]
1470 IF LEN(rh$)>=4 THEN LOCATE 1,7:PRIN
     T STRING$(30+LEN(rh$)," "):GOTO 142
     0                                      [2458]
1480 IF RIGHT$(rh$,1)="." THEN LOCATE 1,
     7:PRINT STRING$(30+LEN(rh$)," ");:G
     OTO 1420                               [9210]
1490 rh=VAL(rh$)                            [EB6E]
1500 IF rh<2 OR rh>32 THEN LOCATE 1,7:PR
     INT STRING$(30," ");:GOTO 1420         [0DA2]
1510 LOCATE 1,9:INPUT"Uebernahmefrequenz
      in Hz :",f1$:IF LEN(f1$)>5 THEN LO
     CATE 1,9:PRINT STRING$(30+LEN(f1$),
     " "):GOTO 1510                         [D746]
1520 IF f1$="" THEN 1560 ELSE x$=f1$:y$=
     RIGHT$(f1$,4):IF ASC(LEFT$(x$,1))<4
     8 OR ASC(LEFT$(x$,1))>57 THEN LOCAT
     E 1,9:PRINT STRING$(30+LEN(f1$)," "
     ):GOTO 1510                            [817C]
1530 x$=LEFT$(y$,1):y$=RIGHT$(f1$,3):IF
     ASC(LEFT$(x$,1))<48 OR ASC(LEFT$(x$
     ,1))>57 THEN LOCATE 1,9:PRINT STRIN
     G$(30+LEN(f1$)," "):GOTO 1510          [421C]
1540 x$=LEFT$(y$,1):y$=RIGHT$(f1$,2):IF
     ASC(LEFT$(x$,1))<48 OR ASC(LEFT$(x$
     ,1))>57 THEN LOCATE 1,9:PRINT STRIN
     G$(30+LEN(f1$)," "):GOTO 1510          [981C]
1550 x$=LEFT$(y$,1):y$=RIGHT$(f1$,1):IF
     ASC(LEFT$(x$,1))<48 OR ASC(LEFT$(x$
     ,1))>57 THEN LOCATE 1,9:PRINT STRIN
     G$(30+LEN(f1$)," "):GOTO 1510          [D61C]
1560 x$=LEFT$(y$,1):IF ASC(x$)<48 OR ASC
     (x$)>57 THEN LOCATE 1,9:PRINT STRIN
     G$(30+LEN(f1$)," "):GOTO 1510          [C8EC]
1570 IF RIGHT$(f1$,1)="." THEN LOCATE 1,
     9:PRINT STRING$(30+LEN(f1$)," "):GO
     TO 1510                                [D792]
1580 IF f1$="" OR VAL(f1$)=0 THEN LOCATE
      1,9:PRINT STRING$(30+LEN(f1$)," ")
     :GOTO 1510                             [E912]
1590 f1=VAL(f1$)                            [3B64]
1600 IF f1<50 OR f1>5000 THEN LOCATE 1,9
     :PRINT STRING$(30+LEN(f1$)," "):GOT
     O 1510                                 [9678]
1610 PRINT                                  [C4EA]
1620 REM 2-weg 6 dB/Okt.                    [066C]
1630 lt26db=1000*rt/(2*3.1415927*f1)        [70E2]
1640 ch26db=1000000/(2*3.1415927*f1*rh)     [31C2]
1650 LOCATE 1,13:PRINT"Zweiwegweiche mit
      6dB/Okt"                              [5154]
1660 LOCATE 1,15:PRINT"Lt = ";:PRINT USI
     NG"####.##";lt26db;:PRINT" mH"         [4D40]
1670 LOCATE 1,17:PRINT"Ch = ";:PRINT USI
     NG"####.##";ch26db;:PRINT " "+CHR$(
     183)+"F"                               [4964]
1680 lt=1000*rt*1.4142/(2*3.1415927*f1)     [1738]
1690 ct=1000000/(2*3.1415927*f1*rt*1.414
     2)                                     [A848]
1700 lh=1000*rh*1.4142/(2*3.1415927*f1)     [E7FA]
1710 ch=1000000/(2*3.1415927*f1*rh*1.414
     2)                                     [C40A]
1720 LOCATE 40,13:PRINT"Zweiwegweiche mi
     t 12dB/Okt."                           [5F6C]
1730 LOCATE 40,15:PRINT"Lt = ";:PRINT US
     ING"####.##";lt;:PRINT" mH"            [8C46]
1740 LOCATE 40,17:PRINT"Ct = ";:PRINT US
     ING"####.##";ct;:PRINT" "+CHR$(183)
     +"F"                                   [FD5A]
1750 LOCATE 40,19:PRINT"Lh = ";:PRINT US
     ING"####.##";lh;:PRINT" mH"            [B022]
1760 LOCATE 40,21:PRINT"Ch = ";:PRINT US
     ING"####.##";ch;:PRINT" "+CHR$(183)
     +"F"                                   [8224]
1770  GOTO 2520                             [5762]
1780 LOCATE 45,9:PRINT STRING$(36," "):L
     OCATE 45,9:PRINT"D R E I W E G W E
     I C H E . . ."                         [0CA6]
1790 LOCATE 1,5:INPUT"Impedanz des Tieft
     oeners<5>: ",rt$                       [ADAA]
1800 IF LEN(rt$)>3 OR rt$=""THEN LOCATE
     1,5:PRINT STRING$(31+LEN(rt$)," ");
     :GOTO 1790                             [5F26]
1810 IF ASC(LEFT$(rt$,1))<48 OR ASC(LEFT
     $(rt$,1))>57 THEN LOCATE 1,5:PRINT
     STRING$(31+LEN(rt$)," ");:GOTO 1790
                                            [825C]
1820 IF LEN(rt$)=3 THEN x$=LEFT$(rt$,2):
     x$=RIGHT$(x$,1):IF x$<>"." THEN LOC
     ATE 1,5:PRINT STRING$(31+LEN(rt$),"
      "):GOTO 1790                          [FC9A]
1830 IF rt$="0" THEN LOCATE 1,5:PRINT ST
     RING$(33," ");:GOTO 1790               [A2EE]
1840 IF LEN(rt$)>=4 THEN LOCATE 1,5:PRIN
     T STRING$(31+LEN(rt$)," "):GOTO 179
     0                                      [269C]
1850 IF RIGHT$(rt$,1)="." THEN LOCATE 1,
     5:PRINT STRING$(31+LEN(rt$)," ");:G
     OTO 1790                               [A054]
1860 rt=VAL(rt$)                            [CEA0]
1870 IF rt<2 OR rt>32 THEN LOCATE 1,5:PR
     INT STRING$(38," ");:GOTO 1790         [0C06]
1880 LOCATE 1,6:INPUT"Impedanz des Mitte
     ltoeners<3>: ",rm$                     [E9EC]
1890 IF LEN(rm$)>3 OR rm$=""THEN LOCATE
     1,6:PRINT STRING$(31+LEN(rm$)," ");
     :GOTO 1880                             [8810]
1900 IF ASC(LEFT$(rm$,1))<48 OR ASC(LEFT
     $(rm$,1))>57 THEN LOCATE 1,6:PRINT
     STRING$(31+LEN(rm$)," ");:GOTO 1880
                                            [BA34]
1910 IF LEN(rm$)=3 THEN x$=LEFT$(rm$,2):
     x$=RIGHT$(x$,1):IF x$<>"." THEN LOC
     ATE 1,6:PRINT STRING$(31+LEN(rm$),"
      "):GOTO 1880                          [8172]
1920 IF rm$="0" THEN LOCATE 1,6:PRINT ST
     RING$(33," ");:GOTO 1880               [9DE2]
1930 IF LEN(rm$)>=4 THEN LOCATE 1,6:PRIN
     T STRING$(31+LEN(rm$)," "):GOTO 188
     0                                      [4882]
1940 IF RIGHT$(rm$,1)="." THEN LOCATE 1,
     6:PRINT STRING$(31+LEN(rm$)," ");:G
     OTO 1880                               [DC3A]
1950 rm=VAL(rm$)                            [1884]
1960 IF rm<2 OR rm>32 THEN LOCATE 1,6:PR
     INT STRING$(38," ");:GOTO 1880         [3CEC]
1970 LOCATE 1,7:INPUT"Impedanz des Hocht
     oeners<5>: ",rh$:PRINT                 [5818]
1980 IF LEN(rh$)>3 OR rh$=""THEN LOCATE
     1,7:PRINT STRING$(31+LEN(rh$)," ");
     :GOTO 1970                             [D1F4]
1990 IF ASC(LEFT$(rh$,1))<48 OR ASC(LEFT
     $(rh$,1))>57 THEN LOCATE 1,7:PRINT
     STRING$(31+LEN(rh$)," ");:GOTO 1970
                                            [182A]
2000 IF LEN(rh$)=3 THEN x$=LEFT$(rh$,2):
     x$=RIGHT$(x$,1):IF x$<>"." THEN LOC
     ATE 1,7:PRINT STRING$(31+LEN(rh$),"
      "):GOTO 1970                          [D644]
2010 IF rh$="0" THEN LOCATE 1,7:PRINT ST
     RING$(33," ");:GOTO 1970               [55C8]
2020 IF LEN(rh$)>=4 THEN LOCATE 1,7:PRIN
     T STRING$(31+LEN(rh$)," "):GOTO 197
     0                                      [175E]
2030 IF RIGHT$(rh$,1)="." THEN LOCATE 1,
     7:PRINT STRING$(31+LEN(rh$)," ");:G
     OTO 1970                               [3816]
2040 rh=VAL(rh$)                            [6B5E]
2050 IF rh<2 OR rh>32 THEN LOCATE 1,7:PR
     INT STRING$(38," ");:GOTO 1970         [64C8]
2060 LOCATE 1,9:INPUT"1. Uebernahmefrequ
```

```
     enz in Hz<2>: ",f1$                        [6B52]
2070 IF f1$="" THEN 2060 ELSE x$=f1$:y$=
     RIGHT$(f1$,4):IF ASC(LEFT$(x$,1))<4
     8 OR ASC(LEFT$(x$,1))>57 THEN LOCAT
     E 1,9:PRINT STRING$(31+LEN(f1$)," "
     ):GOTO 2060                                [EE7A]
2080 x$=LEFT$(y$,1):y$=RIGHT$(f1$,3):IF
     ASC(LEFT$(x$,1))<48 OR ASC(LEFT$(x$
     ,1))>57 THEN LOCATE 1,9:PRINT STRIN
     G$(31+LEN(f1$)," "):GOTO 2060              [9922]
2090 x$=LEFT$(y$,1):y$=RIGHT$(f1$,2):IF
     ASC(LEFT$(x$,1))<48 OR ASC(LEFT$(x$
     ,1))>57 THEN LOCATE 1,9:PRINT STRIN
     G$(31+LEN(f1$)," "):GOTO 2060              [AF22]
2100 x$=LEFT$(y$,1):y$=RIGHT$(f1$,1):IF
     ASC(LEFT$(x$,1))<48 OR ASC(LEFT$(x$
     ,1))>57 THEN LOCATE 1,9:PRINT STRIN
     G$(31+LEN(f1$)," "):GOTO 2060              [A810]
2110 IF LEN(f1$)>=4 THEN x$=LEFT$(y$,1):
     IF ASC(x$)<48 OR ASC(x$)>57 THEN LO
     CATE 1,9:PRINT STRING$(31+LEN(f1$),
     " "):GOTO 2060                             [2350]
2120 IF f1$="" OR VAL(f1$)=0 THEN LOCATE
      1,9:PRINT STRING$(31+LEN(f1$)," ")
     :GOTO 2060                                 [8304]
2130 f1=VAL(f1$)                                [4852]
2140 IF f1<57 OR f1>5000 THEN LOCATE 1,9
     :PRINT STRING$(31+LEN(f1$)," "):GOT
     O 2060                                     [F18A]
2150 LOCATE 1,10:INPUT"2. Uebernahmefreq
     uenz in Hz<2>: ",f2$                       [00E6]
2160 IF f2$="" THEN 2060 ELSE x$=f2$:y$=
     RIGHT$(f2$,4):IF ASC(LEFT$(x$,1))<4
     8 OR ASC(LEFT$(x$,1))>57 THEN LOCAT
     E 1,10:PRINT STRING$(31+LEN(f2$),"
     "):GOTO 2150                               [9FD2]
2170 x$=LEFT$(y$,1):y$=RIGHT$(f2$,3):IF
     ASC(LEFT$(x$,1))<48 OR ASC(LEFT$(x$
     ,1))>57 THEN LOCATE 1,10:PRINT STRI
     NG$(31+LEN(f2$)," "):GOTO 2150             [9E76]
2180 x$=LEFT$(y$,1):y$=RIGHT$(f2$,2):IF
     ASC(LEFT$(x$,1))<48 OR ASC(LEFT$(x$
     ,1))>57 THEN LOCATE 1,10:PRINT STRI
     NG$(31+LEN(f2$)," "):GOTO 2150             [4076]
2190 x$=LEFT$(y$,1):y$=RIGHT$(f2$,1):IF
     ASC(LEFT$(x$,1))<48 OR ASC(LEFT$(x$
     ,1))>57 THEN LOCATE 1,10:PRINT STRI
     NG$(31+LEN(f2$)," "):GOTO 2150             [E276]
2200 IF LEN(f2$)>=4 THEN x$=LEFT$(y$,1):
     IF ASC(x$)<48 OR ASC(x$)>57 THEN LO
     CATE 1,10:PRINT STRING$(31+LEN(f2$)
     ," "):GOTO 2150                            [31A4]
2210 IF f2$="" OR VAL(f2$)=0 THEN LOCATE
      1,10:PRINT STRING$(31+LEN(f2$)," "
     ):GOTO 2150                                [355A]
2220 f2=VAL(f2$)                                [D656]
2230 IF f2<150 OR f2>20000 THEN LOCATE 1
     ,10:PRINT STRING$(31+LEN(f2$)," "):
     GOTO 2150                                  [EF8E]
2240 IF f1>f2 THEN a=f1:f1=f2:f2=a              [907C]
2250 lt36db=1000*rt/(2*3.1415927*f1)            [25E2]
2260 lm36db=1000*rm/(2*3.1415927*f2)            [01CA]
2270 cm36db=1000000/(2*3.1415927*rm*f1)         [47D8]
2280 ch36db=1000000/(2*3.1415927*rh*f2)         [66C8]
2290 LOCATE 1,13:PRINT"Dreiwegweiche mit
      6dB/Okt."                                 [9A7C]
2300 LOCATE 1,15:PRINT"Lt = ";:PRINT USI
     NG"####.##";lt36db;:PRINT" mH"             [CB32]
2310 LOCATE 1,17:PRINT"Lm = ";:PRINT USI
     NG"####.##";lm36db;:PRINT" mH"             [871C]
2320 LOCATE 1,19:PRINT"Cm = ";:PRINT USI
     NG"####.##";cm36db;:PRINT" "+CHR$(1
     83)+"F"                                    [AF30]
2330 LOCATE 1,21:PRINT"Ch = ";:PRINT USI
     NG"####.##";ch36db;:PRINT" "+CHR$(1
     83)+"F"                                    [E010]
2340 REM dreiweg 12dB/okt                       [59F4]
2350 lt=1000*rt*1.4142/(2*3.1415927*f1)         [402E]
2360 ct=1000000/(2*3.1415927*f1*1.4142*r
     t)                                         [BA3E]
2370 cm=1000000/(2*3.1415927*f1*1.4142*r
     m)                                         [E524]
2380 l2=1000*1.4142*rm/(2*3.1415927*f1)         [34A2]
2390 lm=1000*rm*1.4142/(2*3.1415927*f2)         [7F1C]
2400 c2=1000000/(2*3.1415927*f2*rm*1.414
     2)                                         [82A4]
2410 lh=1000*rh*1.4142/(2*3.1415927*f2)         [7DFA]
2420 ch=1000000/(2*3.1415927*f2*rh*1.414
     2)                                         [2B0A]
2430 LOCATE 40,13:PRINT"Dreiwegweiche mi
     t 12dB/Okt."                               [2334]
2440 LOCATE 40,15:PRINT"Lt = ";:PRINT US
     ING"###.##";lt;:PRINT" mH"                 [BCFE]
2450 LOCATE 40,16:PRINT"Ct = ";:PRINT US
     ING"###.##";ct;:PRINT" "+CHR$(183)+
     "F"                                        [F110]
2460 LOCATE 40,17:PRINT"Lm = ";:PRINT US
     ING"###.##";lm;:PRINT" mH"                 [68EA]
2470 LOCATE 40,18:PRINT"Cm = ";:PRINT US
     ING"###.##";cm;:PRINT" "+CHR$(183)+
     "F"                                        [A2FC]
2480 LOCATE 40,19:PRINT"L2 = ";:PRINT US
     ING"###.##";l2;:PRINT" mH"                 [2C06]
2490 LOCATE 40,20:PRINT"C2 = ";:PRINT US
     ING"###.##";c2;:PRINT" "+CHR$(183)+
     "F"                                        [E806]
2500 LOCATE 40,21:PRINT"Lh = ";:PRINT US
     ING"###.##";lh;:PRINT" mH"                 [F0C2]
2510 LOCATE 40,22:PRINT"Ch = ";:PRINT US
     ING"###.##";ch;:PRINT" "+CHR$(183)+
     "F"                                        [06D4]
2520 LOCATE 60,24:PRINT"Neue Berechnung
     <1>"                                       [2E7A]
2530 LOCATE 60,25:PRINT"Zum Hauptmenue<2
     ><2>"                                      [E354]
2540 awb$=INKEY$:IF awb$="" THEN 2540           [BB58]
2550 IF awb$="2" THEN 200                       [A65A]
2560 IF awb$<>"1" THEN IF awb$<>"2" THEN
      LOCATE 60,25:PRINT STRING$(20," ")
     :GOTO 2530                                 [172E]
2570 GOTO 1230                                  [491A]
2580 END                                        [108C]
2590 '******************************           [AA06]
2600 '**       Dritter Teil        **           [1306]
2610 '******************************           [AFF8]
2620 '**       Daten drucken       **           [75A2]
2630 '******************************           [1DFC]
2640 MODE 2:IF ausgabe=1 THEN WINDOW#1,1
     ,80,5,24 ELSE MODE 1:GOSUB 5640            [1632]
2650 IF ausgabe=1 THEN file=1:GOTO 2780
     ELSE file=8                                [A182]
2660 LOCATE 5,10:PRINT"Drucker vorbereit
     en..."                                     [7B64]
2670 PRINT#file,CHR$(15);CHR$(27)CHR$(51
     )CHR$(30);                                 [AD82]
2680 FOR u=1 TO 2000:NEXT:LOCATE 5,10:PR
     INT STRING$(20," ")                        [0AE2]
2690 IF ffff=1 THEN 2720                        [96C0]
2700 LOCATE 5,10:PRINT"Bitte Namen einge
     ben..."                                    [42F8]
2710 LOCATE 5,14:INPUT name$                    [F46E]
2720 rand=INT((40-LEN(name$))/2):PRINT#f
     ile,TAB(rand);:PRINT#file,CHR$(14)C
     HR$(27)CHR$(45)CHR$(1)name$;CHR$(20
     );CHR$(10)                                 [920E]
2730 IF ffff=1 THEN 2770                        [93C0]
2740 FOR u=10 TO 15:LOCATE 1,u:PRINT STR
     ING$(35," "):NEXT:LOCATE 5,24:PRINT
     "Nicht mehr als 25 Zeichen ...":LOC
     ATE 1,5:INPUT"Bezeichnung Tieftoene
     r<2>: ",tief$                              [9FC0]
2750 IF fff=3 THEN LOCATE 1,7:INPUT"Beze
     ichnung Mitteltoener: ",mittel$            [04DE]
2760 LOCATE 1,9:INPUT"Bezeichnung Hochto
     ener<2>: ",hoch$                           [CE30]
2770 FOR u=1 TO 2:PRINT#file,CHR$(10):NE
     XT                                         [94A2]
2780 IF file=1 THEN MODE 2:WINDOW#1,1,80
     ,5,24: GOSUB 5660                          [7B52]
2790 IF ww(1)=0 THEN IF ww(2)=1 THEN 306
     0                                          [2366]
2800 IF ww(1)=1 THEN IF ww(2)=0 THEN 282
     0                                          [A55C]
2810 IF ww(1)=0 THEN IF ww(2)=0 THEN 356
     0                                          [AA60]
2820 '                                          [2C26]
2830 IF file=1 THEN LOCATE 22,2:PRINT"in
      der Anzeige: "CHR$(24)name$CHR$(24
     ); ELSE IF file=1 THEN LOCATE 22,2:
     PRINT"im Druck: "CHR$(24)name$CHR$(
     24); ELSE IF file=8 THEN MODE 2:GOS
     UB 5660:LOCATE 22,2:PRINT"im Druck:
      "CHR$(24)name$CHR$(24);                   [F5CC]
2840 PRINT#file,CHR$(27)CHR$(45)CHR$(1)"
     G E H A E U S E . . .";CHR$(27)CHR$
     (45)CHR$(0)                                [0AD6]
2850 IF file=8 THEN PRINT#file,CHR$(10)
     ELSE PRINT#file                            [3214]
2860 PRINT#file,"V(as) - Wert<3>: ";vas         [A7F2]
2870 PRINT#file,"Q(ts) - Wert<3>: ";qts         [3D2C]
2880 PRINT#file,"Q(tg) - Wert<3>:<2>0.70
     7"                                         [6628]
2890 IF file=8 THEN PRINT#file,CHR$(10)         [94AA]
2900 IF ff=1 THEN PRINT#file:GOTO 2910          [F4E4]
2910 PRINT#file,"Volumen bedaempft : ";:
     PRINT#file,USING"##.##";vg;:PRINT#f
     ile," Liter"                               [526C]
2920 IF file=8 THEN PRINT#file,CHR$(11);
      ELSE PRINT#file                           [6388]
2930 PRINT#file,"Materialstaerke in mm<2
     >: ",mast                                  [6D70]
2940 PRINT#file,"Standhoehe aussen in cm
     : ",auho                                   [BBF6]
2950 PRINT#file,"Tiefe oder Breite in cm
     : ",tibr                                   [7800]
2960 PRINT#file,"Tiefe oder Breite<6>: "
     ,ROUND(br2,2);" cm"                        [99AA]
2970 IF file=8 THEN PRINT#file,CHR$(10);
```

»Boxenkit« (Fortsetzung)

```
     CHR$(10) ELSE PRINT#file:PRINT#file
                                         [610A]
2980 PRINT#file,CHR$(27)CHR$(45)CHR$(1)"
     Zuschnittmasse...";CHR$(27)CHR$(45)
     CHR$(0)                             [789C]
2990 IF file=8 THEN PRINT#file,CHR$(10)
     ELSE PRINT#file                     [911E]
3000 PRINT#file,"2 Stueck ";:PRINT#file,
     USING"###.##";seiw1;:PRINT#file," c
     m * ";:PRINT#file,USING"###.##";sei
     w2;:PRINT#file,"cm"                 [7822]
3010 PRINT#file,"2 Stueck ";:PRINT#file,
     USING"###.##";depl1;:PRINT#file," c
     m * ";:PRINT#file,USING"###.##";dep
     l2;:PRINT#file,"cm"                 [F4D8]
3020 PRINT#file,"2 Stueck ";:PRINT#file,
     USING"###.##";frpl1;:PRINT#file," c
     m * ";:PRINT#file,USING"###.##";frp
     l2;:PRINT#file,"cm"                 [7416]
3030 IF file=1 THEN LOCATE 60,25:PRINT"b
     itte <Taste>...":CALL &BB06:LOCATE
     60,25:PRINT"<17>"                   [2CCE]
3040 FOR u=1 TO 2:PRINT#file,CHR$(10):NE
     XT                                  [6490]
3050 IF ww(2)=0 THEN 200                 [00EC]
3060 IF fff=3 THEN 3270                  [12EA]
3070 PRINT#file,CHR$(27)CHR$(45)CHR$(1)"
     Z W E I W E G W E I C H E . . ."CHR
     $(27)CHR$(45)CHR$(0)                [75B8]
3080 IF file=8 THEN PRINT#file,CHR$(10)
     ELSE PRINT#file                     [B00C]
3090 PRINT#file,"Impedanz des Tieftoener
     s :";rt,tief$                       [39B6]
3100 PRINT#file,"Impedanz des Hochtoener
     s :";rh,hoch$                       [F076]
3110 PRINT#file,"Uebernahmefrequenz in H
     z :";f1                             [32A2]
3120 PRINT#file,CHR$(10)                 [622A]
3130 PRINT#file,CHR$(27)CHR$(45)CHR$(1)"
     Zweiwegweiche mit 6dB/Okt.";CHR$(27
     )CHR$(45)CHR$(0);                   [52AC]
3140 PRINT#file,CHR$(10)                 [F02E]
3150 PRINT#file,"Lt = ";:PRINT#file,USIN
     G"####.##";lt26db;:PRINT#file," mH"
                                         [D3E8]
3160 PRINT#file,"Ch = ";:PRINT#file,USIN
     G"####.##";ch26db;:PRINT#file," uF"
     ;                                   [B218]
3170 PRINT#file,CHR$(10)                 [8F34]
3180 PRINT#file,CHR$(27)CHR$(45)CHR$(1)"
     Zweiwegweiche mit 12dB/Okt.";CHR$(2
     7)CHR$(45)CHR$(0)                   [369A]
3190 IF file=1 THEN PRINT#file           [CAD2]
3200 PRINT#file,"Lt = ";:PRINT#file,USIN
     G"####.##";lt;:PRINT#file," mH"     [FC84]
3210 PRINT#file,"Ct = ";:PRINT#file,USIN
     G"####.##";ct;:PRINT#file," uF"     [EA6E]
3220 PRINT#file,"Lh = ";:PRINT#file,USIN
     G"####.##";lh;:PRINT#file," mH"     [7858]
3230 PRINT#file,"Ch = ";:PRINT#file,USIN
     G"####.##";ch;:PRINT#file," uF"     [C242]
3240 PRINT#file,CHR$(10);CHR$(10)        [7B0C]
3250 IF file=1 THEN LOCATE 60,25:PRINT"b
     itte <Taste>...":CALL &BB06:GOTO 20
     0                                   [89B4]
3260 IF fff=2 THEN 200                   [D478]
3270 PRINT#file,CHR$(27)CHR$(45)CHR$(1)"
     D R E I W E G W E I C H E . . .";CH
     R$(27)CHR$(45)CHR$(0)               [C2FC]
3280 PRINT#file,CHR$(10)                 [AC38]
3290 PRINT#file,"Impedanz des Tieftoener
     s<5>: ";rt,tief$                    [DDFA]
3300 PRINT#file,"Impedanz des Mitteltoen
     ers<3>: ";rm,mittel$                [DFF8]
3310 PRINT#file,"Impedanz des Hochtoener
     s<5>: ";rh,hoch$                    [7BBC]
3320 PRINT#file,"1. Uebernahmefrequenz i
     n Hz<2>: ";f1                       [25B0]
3330 PRINT#file,"2. Uebernahmefrequenz i
     n Hz<2>: " ;f2;                     [FF6C]
3340 PRINT#file,CHR$(10)                 [9C32]
3350 PRINT#file,CHR$(27)CHR$(45)CHR$(1)"
     Dreiwegweiche mit 6dB/Okt."CHR$(27)
     CHR$(45)CHR$(0);                    [DA08]
3360 PRINT#file,CHR$(10)                 [EE36]
3370 PRINT#file,"Lt = ";:PRINT#file,USIN
     G"####.##";lt36db;:PRINT#file," mH"
                                         [44F2]
3380 PRINT#file,"Lm = ";:PRINT#file,USIN
     G"####.##";lm36db;:PRINT#file," mH"
                                         [D4D8]
3390 PRINT#file,"Cm = ";:PRINT#file,USIN
     G"####.##";cm36db;:PRINT#file," uF"
                                         [0AC2]
3400 PRINT#file,"Ch = ";:PRINT#file,USIN
     G"####.##";ch36db;:PRINT#file," uF"
     ;                                   [D414]
3410 PRINT#file,CHR$(10)                 [0F2E]
3420 IF file=1 THEN LOCATE 60,25:PRINT"b
     itte <Taste>...":CALL &BB06:LOCATE
     60,25:PRINT"<17>"                   [34D4]
3430 PRINT#file,CHR$(27)CHR$(45)CHR$(1)"
     Dreiwegweiche mit 12dB/Okt."CHR$(27
     )CHR$(45)CHR$(0);                   [1260]
3440 PRINT#file,CHR$(10)                 [9C34]
3450 PRINT#file,"Lt = ";:PRINT#file,USIN
     G"###.##";lt;:PRINT#file," mH"      [BA4C]
3460 PRINT#file,"Ct = ";:PRINT#file,USIN
     G"###.##";ct;:PRINT#file," uF"      [4A36]
3470 PRINT#file,"Lm = ";:PRINT#file,USIN
     G"###.##";lm;:PRINT#file," mH"      [EB34]
3480 PRINT#file,"Cm = ";:PRINT#file,USIN
     G"###.##";cm;:PRINT#file," uF"      [311E]
3490 PRINT#file,"L2 = ";:PRINT#file,USIN
     G"###.##";l2;:PRINT#file," mH"      [8B4C]
3500 PRINT#file,"C2 = ";:PRINT#file,USIN
     G"###.##";c2;:PRINT#file," uF"      [6324]
3510 PRINT#file,"Lh = ";:PRINT#file,USIN
     G"###.##";lh;:PRINT#file," mH"      [F816]
3520 PRINT#file,"Ch = ";:PRINT#file,USIN
     G"###.##";ch;:PRINT#file," uF";     [6276]
3530 PRINT#file,CHR$(10);CHR$(10)        [A410]
3540 IF file=1 THEN LOCATE 60,25:PRINT"b
     itte <Taste>...":CALL &BB06:fff=0:G
     OTO 200                             [556A]
3550 GOTO 200                            [72B0]
3560 CLS:LOCATE 10,10:PRINT"K e i n e<3>
     D a t e n . . ."                    [22FA]
3570 LOCATE 65,24:PRINT"Bitte <ENTER>...
     "                                   [B2D2]
3580 CALL &BB06:GOTO 200                 [8CC2]
3590 '********************************   [C708]
3600 '**       Vierter  Teil        **   [4E0E]
3610 '********************************   [A0FA]
3620 '**    Schaltplaene 2,3-weg    **   [81DE]
3630 '********************************   [9EFE]
3640 '**    und Hardcopy-Routine    **   [6060]
3650 '**      fuer Epson-Drucker    **   [D922]
3660 '**    (c) by Happy-Computer   **   [EF56]
3670 '********************************   [8306]
3680 MODE 1:GOSUB 5640:LOCATE 8,5:PRINT
     CHR$(24);STRING$(24," ");CHR$(24)   [1894]
3690 LOCATE 8,6:PRINT CHR$(24);"<6>SCHAL
     TPLAENE<6>";CHR$(24)                [DBD2]
3700 LOCATE 8,7:PRINT CHR$(24);STRING$(2
     4," ");CHR$(24)                     [FED8]
3710 LOCATE 8,10:PRINT"2-weg mit 6dB/Okt
     ......1"                            [7B7A]
3720 LOCATE 8,12:PRINT"2-weg mit 12dB/Ok
     t.....2"                            [8A80]
3730 LOCATE 8,14:PRINT"3-weg mit 6dB/Okt
     ......3"                            [A98C]
3740 LOCATE 8,16:PRINT"3-weg mit 12dB/Ok
     t.....4"                            [5092]
3750 LOCATE 8,18:PRINT"Zum Hauptmenue...
     ......0"                            [DCCA]
3760 LOCATE 8,23:PRINT"Ihre Wahl bitte..
     ......?"                            [506E]
3770 wahl$=INKEY$:IF wahl$="" THEN 3770  [4238]
3780 IF wahl$="0" THEN RETURN ELSE LOCAT
     E 1,20:INPUT"Ist dieser Schaltplan
     auf der Diskette<2>im aktiven Laufw
     erk vorhanden (j/n) ";a$:IF a$<>"j"
     THEN RETURN                         [F128]
3790 IF wahl$="1" THEN MODE 2:INK 1,13:L
     OAD"2weg6db.grf",&C000:GOSUB 5660:G
     OSUB 5680:GOTO 3840                 [A6C4]
3800 IF wahl$="2" THEN MODE 2:INK 1,13:L
     OAD"2weg12db.grf",&C000:GOSUB 5660:
     GOSUB 5830:GOTO 3840                [C60A]
3810 IF wahl$="3" THEN MODE 2:INK 1,13:L
     OAD"3weg6db.grf",&C000:GOSUB 5660:G
     OSUB 5930:GOTO 3840                 [4EB8]
3820 IF wahl$="4" THEN MODE 2:INK 1,13:L
     OAD"3weg12db.grf",&C000:GOSUB 5660:
     GOSUB 6030:GOTO 3840                [E906]
3830 IF wahl$<>"0" OR wahl$<>"1" OR wahl
     $<>"2" OR wahl$<>"3" OR wahl$<>"4"
     THEN 3770                           [7042]
3840 LOCATE 1,25:PRINT CHR$(24);"<2>Hard
     copy.......1<5>Menue..........0<20>
     Ihre Wahl bitte...?<2>";CHR$(24);   [FF9E]
3850 INK 1,1                             [8600]
3860 rwahl$=INKEY$:IF rwahl$="" THEN 386
     0                                   [CF00]
3870 IF rwahl$="1" THEN GOSUB 3900:GOTO
     3610                                [E69A]
3880 IF rwahl$="0" THEN GOTO 3610        [A64E]
3890 IF rwahl$<>"1" OR rwahl$<>"0" THEN
     3860                                [514A]
3900 '*** HARDCOPY ***                   [4352]
3910 LOCATE 1,25:PRINT STRING$(80," ");  [DFDE]
3920 LOCATE 1,25:PRINT"Bitte Drucker ber
     eitmachen dann <TASTE>":CALL &BB06  [F7CC]
3930 LOCATE 1,25:PRINT STRING$(80," ");  [B3E2]
3940 |HARDCOPY:RETURN                    [16C0]
3950 i=40960                             [B134]
3960 READ wert$:IF wert$="ende" THEN 398
     0                                   [4786]
3970 POKE i,VAL("&"+wert$):i=i+1:GOTO 39
     60                                  [0C20]
3980 CALL &A000:RETURN                   [0682]
```

```
3990 DATA 01,09,A0,21,17,A0,C3,D1,BC,0E,
     A0,C3,1B,A0,48,41,52,44,43,4F,50,D9
     ,00,00,00,00,00,CD,0B,BC,22,4C,A1,C
     D,11,BC,32,4E,A1,21,5F,A1,3D,28,09,
     21,57,A1,3D,28,03,21,63,A1,01,08,00
     ,11,4F,A1,ED,B0,ED,73,6E,A1,3A,4E,A
     1,87,FE,01,CE,00,87,32,66,A1,0E,0D  [1B44]
4000 DATA CD,37,A1,0E,0A,CD,37,A1,3E,02,
     CD,AD,A0,3E,7F,32,65,A1,2A,4C,A1,7C
     ,F6,C0,67,06,1D,05,20,05,3E,78,32,6
     5,A1,04,C5,CD,C2,A0,3A,4E,A1,FE,02,
     CC,C2,A0,11,00,08,A7,ED,52,7C,F6,3F
     ,3C,28,08,11,B0,3F,19,7C,F6,F8,67,0
     E,0A,CD,37,A1,0E,0D,CD,37,A1,E5,3E  [356E]
4010 DATA 42,CD,1E,BB,C2,47,A1,E1,C1,10,
     C0,3E,03,21,70,A1,4E,23,06,00,09,3D
     ,20,F8,46,23,4E,23,CD,37,A1,10,F9,C
     9,E5,21,70,A1,3A,4E,A1,FE,02,28,05,
     4E,23,06,00,09,CD,B8,A0,E1,3A,4E,A1
     ,FE,02,06,28,28,02,06,50,C5,E5,11,6
     7,A1,3E,07,ED,A0,2B,01,00,08,09,30  [DDFE]
4020 DATA 0A,01,50,C0,09,47,7C,E6,C7,67,
     78,3D,20,EA,21,4F,A1,3A,66,A1,47,C5
     ,11,67,A1,06,07,1A,13,A6,FE,01,3F,C
     B,11,10,F6,3A,65,A1,A1,4F,CD,37,A1,
     3A,4E,A1,A7,CC,37,A1,C1,23,10,DD,E1
     ,C1,7C,E6,F8,4F,23,7C,E6,07,B1,67,1
     0,AB,C9,79,CD,2B,BD,D8,C5,E5,3E,42  [6D28]
4030 DATA CD,1E,BB,E1,C1,28,F0,ED,7B,6E,
     A1,C9,00,00,00,00,00,00,00,00,00,00
     ,00,80,40,20,10,08,04,02,01,88,44,2
     2,11,AA,55,00,00,00,00,00,00,00,00,
     00,00,00                            [76A6]
4040 DATA 04,1B,4C,40,01,04,1B,4B,40,01,
     03,1B,41,07,03,1B,41,0C             [7C06]
4050 DATA ende                           [F37E]
4060 '******************************     [86FA]
4070 '**      Fuenfter  Teil        **   [408E]
4080 '******************************     [6CFE]
4090 '**   Programm  beenden        **   [7F64]
4100 '******************************     [4CF0]
4110 LOCATE 10,25:PRINT"wirklich beenden
      j/n "                              [A6A4]
4120 wahl1$=INKEY$:IF wahl1$="" THEN 412
     0                                   [D2C0]
4130 wahl1$=UPPER$(wahl1$)               [553C]
4140 IF wahl1$="N" THEN 200 ELSE IF wahl
     1$="J" THEN FOR u=1 TO 1000:NEXT:CL
     S:END                               [31CC]
4150 IF wahl1$<>"N" THEN IF wahl1$<>"J"
     THEN 4120                           [A6F2]
4160 '******************************     [CAFC]
4170 '**      Sechster  Teil        **   [7A94]
4180 '******************************     [2500]
4190 '**        Hinweise zu         **   [61D0]
4200 '**  erforderlichen  Daten     **   [299A]
4210 '******************************     [2DF4]
4220 WINDOW#1,1,80,4,25:CLS#1            [FBD0]
4230 LOCATE 1,6:PRINT"Unter dem Menuepun
     kt '2' werden Sie da- zu<2>aufgefor
     dert, einen V(as)-Wert ein- zugeben
     ."                                  [3500]
4240 PRINT                               [AEEE]
4250 LOCATE 1,10:PRINT"Dieser V(as)-Wert
      ist ein Mass fuer die Nachgiebigke
     it der Lautsprechermembrane und gib
     t an, welche Menge<2>an<2>Luft<2>in
      einem Behaelter<2>gleicher<2>Grund
     flaeche ebenso nachgiebig ist, wie<
     2>die Membran in ihrer Aufhaengung.
     "                                   [FA48]
4260 PRINT                               [B8F2]
4270 LOCATE 1,17:PRINT"Hersteller<2>gute
     r<2>Lautsprechersysteme<2>geben die
     sen Wert in ihren Datenblaet-tern m
     it an oder teilen ihn<2>Ihnen<2>auf
     <3>Anfrage gerne mit."              [2C84]
4280 LOCATE 25,24:PRINT"bitte <ENTER>":C
     ALL &BB06                           [BB00]
4290 CLS#1                               [CA4A]
4300 LOCATE 1,6:PRINT"Der angeforderte Q
     (ts)-Wert ist<2>ebenso in guten Dat
     enblaettern aufgefuehrt. Er ist ein
     e dimensionslose rein technische Gr
     oesse, deren Ermittlung aufwendig i
     st und hier nicht naeher erlaeutert
      werden soll."                      [094C]
4310 PRINT                               [D5EA]
4320 LOCATE 1,13:PRINT"Als<2>Anhaltspunk
     t<2>sei<2>angemerkt, dass bessere L
     autsprecher Q(ts) Werte<2>unter 0.3
     5 besitzen und gute sogar Werte<2>u
     n- ter 0.26 und weniger aufweisen k
     oennen. Auch diesen Wert erhalten<2
     >Sie<2>aus Da- tenblaettern oder au
     f Anfrage."                         [4030]
4330 LOCATE 25,24:PRINT"Bitte <ENTER>":C
     ALL &BB06:RETURN                    [DDEC]
4340 '******************************     [B4FC]
4350 '**      Siebter  Teil         **   [28EE]
4360 '******************************     [2300]
4370 '**    Daten  speichern        **   [D6D4]
4380 '******************************     [C904]
4390 CLS:GOSUB 5640                      [20F6]
4400 LOCATE 11,5:PRINT STRING$(17,CHR$(1
     43)):LOCATE 11,6:PRINT CHR$(24)" DA
     TEN SPEICHERN "CHR$(24):LOCATE 11,7
     :PRINT STRING$(17,CHR$(143))        [CC34]
4410 LOCATE 5,10:PRINT"Auf welchem Laufw
     erk ? (A/B)"                        [6424]
4420 aa$=INKEY$:LOCATE 20,15:PRINT"DRIVE
     ..."UPPER$( aa$):aa$=UPPER$(aa$):IF
      aa$="" THEN 4410 ELSE IF aa$="A" T
     HEN |A ELSE |B                      [AF58]
4430 FOR u=1 TO 2000:NEXT:LOCATE 5,10:PR
     INT STRING$(29," "):LOCATE 5,15:PRI
     NT STRING$(29," ")                  [D4F8]
4440 LOCATE 8,10:PRINT"Bitte Namen einge
     ben..."                             [4004]
4450 LOCATE 8,14:INPUT name$:name$=name$
     +".box"                             [A748]
4460 OPENOUT name$                       [DCFA]
4470 PRINT#9,name$                       [FD92]
4480 PRINT#9,fff                         [306E]
4490 FOR u=10 TO 16:LOCATE 5,u:PRINT STR
     ING$(34," "):NEXT:LOCATE 5,24:PRINT
     "Nicht mehr als 25 Zeichen ...":LOC
     ATE 1,10:INPUT"Bezeichnung Tieftoen
     er<2>: ",tief$                      [0D28]
4500 IF fff=3 THEN LOCATE 1,12:INPUT"Bez
     eichnung Mitteltoener: ",mittel$    [572C]
4510 LOCATE 1,14:INPUT"Bezeichnung Hocht
     oener<2>: ",hoch$                   [F97E]
4520 GOTO 4560                           [6026]
4530 'IF ww(1)=0 THEN IF ww(2)=1 THEN 45
     50                                  [92B2]
4540 IF ww(1)=1 THEN IF ww(2)=0 THEN 456
     0                                   [EE68]
4550 IF ww(1)=0 THEN IF ww(2)=0 THEN 498
     0                                   [7C74]
4560 PRINT#9,tief$:PRINT#9,mittel$:PRINT
     #9,hoch$                            [AECE]
4570 PRINT#9,ww(1)                       [4AEA]
4580 PRINT#9,ww(2)                       [B6EE]
4590 PRINT#9,vas                         [02A2]
4600 PRINT#9,qts                         [38AE]
4610 PRINT#9,qtg                         [EF98]
4620 PRINT#9,vg                          [25BC]
4630 PRINT#9,mast                        [6A6E]
4640 PRINT#9,auho                        [AD60]
4650 PRINT#9,tibr                        [8A6A]
4660 PRINT#9,br2                         [DE16]
4670 PRINT#9,frpl1:PRINT#9,frpl2         [E840]
4680 PRINT#9,depl1:PRINT#9,depl2         [C106]
4690 PRINT#9,frpl1:PRINT#9,frpl2         [3244]
4700 '                                   [9624]
4710 PRINT#9,rt                          [04CE]
4720 PRINT#9,rh                          [12B8]
4730 PRINT#9,f1                          [6A34]
4740 PRINT#9,lt26db                      [5B24]
4750 PRINT#9,ch26db                      [4BFC]
4760 PRINT#9,ct                          [2DBA]
4770 PRINT#9,lh                          [02B6]
4780 PRINT#9,ch                          [55A6]
4790 '                                   [BF36]
4800 PRINT#9,rt                          [51CE]
4810 PRINT#9,rm                          [45C2]
4820 PRINT#9,rh                          [C9BA]
4830 PRINT#9,f1                          [8136]
4840 PRINT#9,f2                          [C93A]
4850 PRINT#9,lt36db                      [7D2A]
4860 PRINT#9,lm36db                      [CD1E]
4870 PRINT#9,cm36db                      [800E]
4880 PRINT#9,ch36db                      [6E06]
4890 PRINT#9,lt                          [2DD4]
4900 PRINT#9,ct                          [2BB2]
4910 PRINT#9,lm                          [62B8]
4920 PRINT#9,cm                          [4DA8]
4930 PRINT#9,12                          [A046]
4940 PRINT#9,ch                          [73A2]
4950 PRINT#9,lh                          [70B6]
4960 PRINT#9,ch                          [13A6]
4970 CLOSEOUT:GOTO 200                   [B30E]
4980 FOR u=10 TO 22:LOCATE 5,u:PRINT STR
     ING$(34," "):NEXT:LOCATE 10,10:PRIN
     T"K e i n e<3>D a t e n . . ."      [D51C]
4990 LOCATE 65,24:PRINT"Bitte <ENTER>...
     "                                   [02E0]
5000 CALL &BB06:CLOSEOUT:GOTO 200        [70FC]
5010 '******************************     [A2F2]
5020 '**       Achter  Teil         **   [55F6]
5030 '******************************     [A0F6]
5040 '** a. Daten einlesen und      **   [CC9A]
5050 '** b.          ausgeben       **   [7426]
5060 '******************************     [21FC]
5070 '** b.1  Bildschirmausgabe     **   [B02A]
5080 '** b.2  Druckerausgabe        **   [D08C]
5090 '******************************     [6B02]
```

»Boxenkit« (Fortsetzung)

```
5100 CLS:GOSUB 5640                                           [76E2]
5110 LOCATE 11,5:PRINT STRING$(17,CHR$(1
     43)):LOCATE 11,6:PRINT CHR$(24)" DA
     TEN<2>EINLESEN "CHR$(24):LOCATE 11,
     7:PRINT STRING$(17,CHR$(143))                            [19D6]
5120 LOCATE 1,15:PRINT STRING$(39," ");:
     LOCATE 20,24:PRINT STRING$(19," ");
                                                              [76B6]
5130 GOSUB 5640                                               [24B0]
5140 name$=qwe$                                               [5EBA]
5150 LOCATE 5,10:PRINT"Von welchem Laufw
     erk ? (A/B)"                                             [7F56]
5160 d$=INKEY$:LOCATE 20,15:PRINT"DRIVE.
     .."UPPER$(d$):d$=UPPER$(d$):IF d$="
     " THEN 5160 ELSE IF d$="A" THEN |A
     ELSE |B                                                  [EEBA]
5170 WINDOW#3,1,40,5,25:CLS#3                                 [6BDC]
5180 LOCATE 8,11:PRINT"Directory auslese
     n.........1"                                             [6036]
5190 LOCATE 8,13:PRINT"Zum Hauptmenue...
     ..........0"                                             [1C30]
5200 LOCATE 8,19:PRINT"Ihre Wahl bitte..
     ..........?"                                             [22D6]
5210 a$=INKEY$:IF a$="" THEN 5210                             [63E8]
5220 a=INT(VAL(a$)):IF a<0 OR a>1 THEN 5
     170                                                      [2A3E]
5230 IF a=0 THEN RETURN                                       [136C]
5240 MODE 2:GOSUB 5660                                        [C01A]
5250 IF a=1 THEN PRINT:PRINT:PRINT:PRINT
     :CAT                                                     [749A]
5260 IF VAL(a$)=0 THEN RETURN                                 [0822]
5270 LOCATE 5,5:INPUT"Name des gewuensch
     ten Files (ohne Extension) : ",name
     $:IF name$=swe$ THEN RETURN ELSE na
     me$=name$+".box"                                         [CA60]
5280 name$=name$:IF name$="" THEN RETURN
                                                              [0E68]
5290 WINDOW#1,1,80,7,25:CLS#1:LOCATE 1,5
     :PRINT STRING$(79," "):LOCATE 1,5:P
     RINT"Gelesen wird : ";name$:WINDOW
     SWAP 0,1                                                 [AC4C]
5300 OPENIN name$                                             [542C]
5310 WHILE NOT EOF                                            [DEDA]
5320 INPUT#9,name$:INPUT#9,fff                                [1596]
5330 INPUT#9,tief$:INPUT#9,mittel$:INPUT
     #9,hoch$                                                 [BDD8]
5340 INPUT#9,ww(1):INPUT#9,ww(2)                              [8B6E]
5350 INPUT#9,vas:INPUT#9,qts:INPUT#9,qtg
                                                              [D92E]
5360 INPUT#9,vg:INPUT#9,mast:INPUT#9,auh
     o:INPUT#9,tibr                                           [C7D8]
5370 INPUT#9,br2:INPUT#9,seiw1:INPUT#9,s
     eiw2:INPUT#9,depl1:INPUT#9,depl2:IN
     PUT#9,frpl1:INPUT#9,frpl2                                [9588]
5380 '                                                        [272E]
5390 INPUT#9,rt:INPUT#9,rh:INPUT#9,f1                         [FC08]
5400 INPUT#9,lt26db:INPUT#9,ch26db:INPUT
     #9,ct:INPUT#9,lh:INPUT#9,ch                              [5B8C]
5410 INPUT#9,rt:INPUT#9,rmh:INPUT#9,rh:I
     NPUT#9,f1:INPUT#9,f2                                     [2300]
5420 INPUT#9,lt36db:INPUT#9,lm36db:INPUT
     #9,cm36db:INPUT#9,ch36db                                 [1812]
5430 INPUT#9,lt:INPUT#9,ct:INPUT#9,lm:IN
     PUT#9,cm:INPUT#9,l2:INPUT#9,ch:INPU
     T#9,lh:INPUT#9,ch                                        [8F54]
5440 'IF a=2 THEN ausgabe=0:ffff=1:GOTO
     5490                                                     [B2D8]
5450 ausgabe=1                                                [9568]
5460 WEND:CLOSEIN                                             [CAC8]
5470 MODE 1:GOSUB 5640                                        [801E]
5480 LOCATE 11,5:PRINT STRING$(17,CHR$(1
     43)):LOCATE 11,6:PRINT CHR$(24)" DA
     TEN<2>EINLESEN "CHR$(24):LOCATE 11,
     7:PRINT STRING$(17,CHR$(143))                            [74EA]
5490 LOCATE 8,11:PRINT"Ausgabe auf Bilds
     chirm.....1"                                             [14CE]
5500 LOCATE 8,13:PRINT"Ausgabe auf Druck
     er........2"                                             [2776]
5510 LOCATE 8,15:PRINT"Zum Hauptmenue...
     ..........0"                                             [7C2C]
5520 LOCATE 8,19:PRINT"Ihre Wahl bitte..
     ..........?"                                             [20E0]
5530 a$=INKEY$:IF a$="" THEN 5530 ELSE a
     =INT(VAL(a$))                                            [07F4]
5540 IF a<0 OR a>2 THEN CLS#3:GOTO 5490                       [7350]
5550 IF a=0 THEN RETURN                                       [A676]
5560 IF a=1 THEN ausgabe=1 ELSE ausgabe=
     0:ffff=1                                                 [E822]
5570 GOSUB 2640:RETURN                                        [BEEE]
5580 '*******************************                         [A80A]
5590 '**     Neunter  Teil         **                         [0622]
5600 '*******************************                         [2BFC]
5610 '**     Kopfzeile MODE 1      **                         [7674]
5620 '**     Kopfzeile MODE 2      **                         [A278]
5630 '*******************************                         [A102]
5640 LOCATE 1,1:PRINT STRING$(40,CHR$(15
     4));"Boxenkit 4.3<4>(C) 1986 by<2>G
      A L A C O";:PRINT STRING$(40,CHR$(
     154));                                                   [5F5E]
5650 RETURN                                                   [91A0]
5660 PRINT STRING$(80,CHR$(154));"Boxenk
     it 4.3<44>(C) 1986 by<2>G A L A C O
     ";:PRINT STRING$(80,CHR$(154));                          [7232]
5670 RETURN                                                   [8DA4]
5680 '*******************************                         [540C]
5690 '**      Zehnter  Teil        **                         [5C22]
5700 '*******************************                         [83FE]
5710 '**  Daten in Schaltplaene    **                         [C766]
5720 '**        einbinden          **                         [C712]
5730 '*******************************                         [1F04]
5740 '                                                        [BB2E]
5750 '*******************************                         [BD08]
5760 '**         2weg 6db          **                         [DF24]
5770 '*******************************                         [2B0C]
5780 IF fff=2 OR ffff=2 THEN 5790 ELSE 5
     820                                                      [2B50]
5790 IF name$<>"" THEN LOCATE 20,2:PRINT
     "zu den Daten von : "CHR$(24)name$C
     HR$(24); ELSE LOCATE 20,2:PRINT CHR
     $(24)"zur momentanen Berechnung"CHR
     $(24);                                                   [B526]
5800 LOCATE 60,9:PRINT"lt =";:PRINT USIN
     G"###.##";lt26db;:PRINT" mH"                             [860A]
5810 LOCATE 60,10:PRINT"ch =";:PRINT USI
     NG"###.##";ch26db;:PRINT " "+CHR$(1
     83)+"F"                                                  [617A]
5820 RETURN                                                   [059E]
5830 '*******************************                         [B106]
5840 '**         2weg 12db         **                         [803C]
5850 '*******************************                         [B70A]
5860 IF fff=2 OR ffff=2 THEN 5870 ELSE 5
     920                                                      [C04E]
5870 IF name$<>"" THEN LOCATE 20,2:PRINT
     "zu den Daten von : "CHR$(24)name$C
     HR$(24); ELSE LOCATE 20,2:PRINT CHR
     $(24)"zur momentanen Berechnung"CHR
     $(24);                                                   [6424]
5880 LOCATE 60,9:PRINT"lt =";:PRINT USIN
     G"###.##";lt;:PRINT" mH"                                 [D3BE]
5890 LOCATE 60,10:PRINT"ct =";:PRINT USI
     NG"###.##";ct;:PRINT " "+CHR$(183)+
     "F"                                                      [7D5E]
5900 LOCATE 60,11:PRINT "lh =";:PRINT US
     ING"###.##";lh;:PRINT" mH"                               [0512]
5910 LOCATE 60,12:PRINT"ch =";:PRINT USI
     NG"###.##";ch;:PRINT " "+CHR$(183)+
     "F"                                                      [DD24]
5920 RETURN                                                   [A6A0]
5930 '*******************************                         [CF08]
5940 '**         3weg 6db          **                         [3126]
5950 '*******************************                         [510C]
5960 IF fff=3 OR ffff=3 THEN 5980 ELSE 6
     020                                                      [3B48]
5970 IF name$<>"" THEN LOCATE 20,2:PRINT
     "zu den Daten von : "CHR$(24)name$C
     HR$(24); ELSE LOCATE 20,2:PRINT CHR
     $(24)"zur momentanen Berechnung"CHR
     $(24);                                                   [BF26]
5980 LOCATE 66,9:PRINT"lt =";:PRINT USIN
     G"###.##";lt36db;:PRINT" mH"                             [F72A]
5990 LOCATE 66,10:PRINT"lm =";:PRINT US
     ING"###.##";lm36db;:PRINT" mH"                           [8DA0]
6000 LOCATE 66,11:PRINT"cm =";:PRINT USI
     NG"###.##";cm36db;:PRINT " "+CHR$(1
     83)+"F"                                                  [EF8E]
6010 LOCATE 66,12:PRINT "ch =";:PRINT US
     ING"###.##";ch36db;:PRINT " "+CHR$(
     183)+"F"                                                 [CDBE]
6020 RETURN                                                   [C590]
6030 '*******************************                         [81F8]
6040 '**         3weg 12db         **                         [4F30]
6050 '*******************************                         [5BFC]
6060 IF fff=3 OR ffff=3 THEN 6080 ELSE 6
     160                                                      [4132]
6070 IF name$<>"" THEN LOCATE 20,2:PRINT
     "zu den Daten von : "CHR$(24)name$C
     HR$(24); ELSE LOCATE 20,2:PRINT CHR
     $(24)"zur momentanen Berechnung"CHR
     $(24);                                                   [3E16]
6080 LOCATE 66,9:PRINT"lt =";:PRINT USIN
     G"###.##";lt;:PRINT" mH"                                 [AABC]
6090 LOCATE 66,10:PRINT "ct =";:PRINT US
     ING"###.##";ct;:PRINT " "+CHR$(183)
     +"F"                                                     [339C]
6100 LOCATE 66,11:PRINT "lm =";:PRINT US
     ING"###.##";lm;:PRINT" mH"                               [5924]
6110 LOCATE 66,12:PRINT"cm =";:PRINT USI
     NG"###.##";cm;:PRINT " "+CHR$(183)+
     "F"                                                      [7136]
6120 LOCATE 66,13:PRINT"l2 =";:PRINT USI
     NG"###.##";l2;:PRINT" mH"                                [DC00]
6130 LOCATE 66,14:PRINT"c2 =";:PRINT USI
     NG"###.##";c2;:PRINT " "+CHR$(183)+
     "F"                                                      [E352]
6140 LOCATE 66,15:PRINT"lh =";:PRINT USI
     NG"###.##";lh;:PRINT" mH"                                [E6E0]
6150 LOCATE 66,16:PRINT "ch =";:PRINT US
     ING"###.##";ch;:PRINT " "+CHR$(183)
     +"F"                                                     [3872]
6160 RETURN                                                   [049A]
```

»Boxenkit« (Schluß)

Aktienkurse für jedermann

»Shares« hilft jedem, die Kursentwicklung seiner diversen Aktien im Auge zu behalten und damit immer eine Übersicht zu haben.

Immer mehr Menschen legen ihren Spargroschen nicht unters Kopfkissen, sondern setzen auf die eventuell hohe Rendite von Wertpapieren. Im Laufe der Zeit sammelt sich so ein ziemlicher Haufen Papier an, der verwaltet sein will. Das übernehmen zwar im allgemeinen Banken, man selbst möchte sich aber oft auch einen kurzfristigen Überblick verschaffen. »Shares« nimmt Ihnen nun diese Arbeit ab und verleiht Ihnen Übersicht über die Kursentwicklung jedes einzelnen Papiers. Die Bedienung gestaltet sich dank der weitgehenden Joysticksteuerung sehr benutzerfreundlich. Eingaben erleichtert außerdem eine deutsche Tastaturbelegung. Die Arbeit mit Shares erfolgt über ein Menü, dessen Unterpunkte mit Hilfe eines Pfeilsymbols anzuwählen sind. Dazu richten Sie den Pfeil mit dem Joystick auf den Punkt und drücken die Feuertaste.

– Laden vorhandener Daten

Lädt den Inhalt der Datei »SHARES.DAT« zur Bearbeitung in den Arbeitsspeicher. Sie besteht aus bis zu zehn Datensätzen für je ein Wertpapier.

– Dateieninhaltsverzeichnis

Zeigt nicht etwa Directory des angeschlossenen Massenspeichers an, sondern für welche Wertpapiere sich Daten im Speicher befinden, und wieviele Kursnotierungen für jedes Wertpapier vorliegen.

– Neuanlage einer Wertpapierdatei

Solange nicht bereits zehn Wertpapieradressen im Speicher sind, läßt sich eine Datei für ein weiteres Wertpapier neu einrichten. Dazu müssen Sie den Namen des Wertpapieres, den Zeitpunkt der ersten Kursnotierung und diese Kursnotierung selbst eingeben.

– Aktualisierung einer Wertpapierdatei

Geben Sie neue Kursnotierungen für ein bereits gespeichertes Wertpapier ein. Wählen Sie das gewünschte Wertpapier mit dem Joystick, geben Sie den Zeitpunkt und die neue Kursnotierung an und legen Sie fest, ob der Kurs in die Kursstatistik des Wertpapiers einbezogen werden soll.

– Kursdaten eines Wertpapiers

Ausgabe sämtlicher gespeicherter Kursnotierungen eines Wertpapiers.

– Kursentwicklung eines Wertpapiers

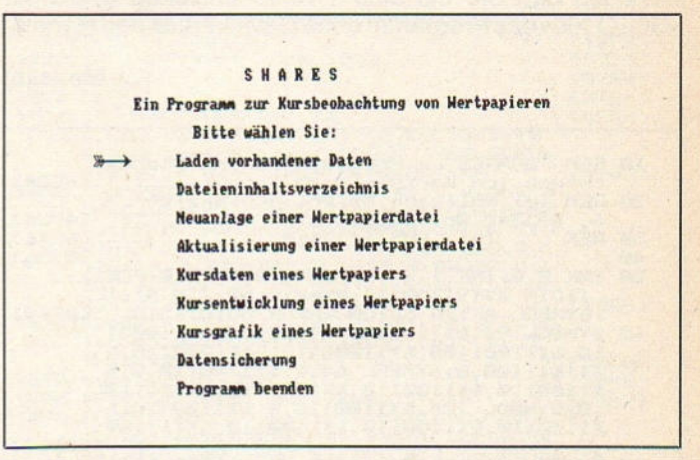
Die Ausgabe beschränkt sich auf die Kursnotierungen, bei deren Eingabe Sie angegeben haben, daß sie in die Kursstatistik einfließen sollen. Zu jeder dieser Notierungen erscheint außerdem die prozentuale Veränderung zur vorhergehenden und zur ersten gespeicherten Kursangabe.

– Kursgrafik eines Wertpapiers

Einen besonders deutlichen Überblick über die Kursentwicklung eines Papiers bietet die grafische Darstellung auf dem Monitor, die sämtliche gespeicherten Kursnotierungen einbezieht. Durch eine Rechts- oder Linksbewegung des Joysticks können Sie jeden Punkt der Kurve (entspricht einer Kursnotierung) mit Hilfe eines kreisförmigen Cursors ansteuern, wodurch Shares den Zeitpunkt und Kurswert des jeweiligen Kurvenpunkts am unteren Rand des Bildschirms anzeigt. Durch Druck der Feuertaste gelangen Sie zurück ins Hauptmenü.

– Datensicherung

Damit Sie nicht nach jedem Start des Programms alle Daten neu eingeben müssen, lassen sich die Daten aller Wertpapiere als Datei »SHARES.DAT« speichern.

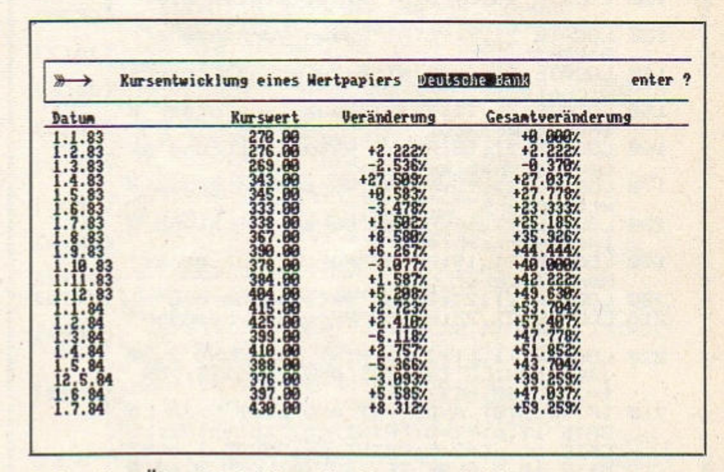


Das Hauptmenü von »Share«

Kursdaten eines Wertpapiers — General Electric Co. — Ausgabeende

Datum	Kurswert	Kommentar
1.1.85	166.75	Tiefstkurs
1.2.85	178.00	
1.3.85	178.00	
1.4.85	189.00	
1.5.85	191.65	
1.6.86	183.00	
1.7.86	194.00	
1.8.86	203.00	
1.9.86	199.00	
1.11.86	245.00	Höchstkurs

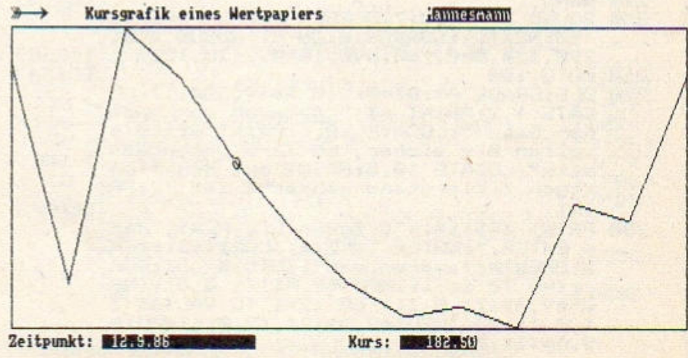
Genaue Daten zu jedem Wertpapier



Kursentwicklung eines Wertpapiers — Deutsche Bank — enter ?

Datum	Kurswert	Veränderung	Gesamtveränderung
1.1.83	270.00		+0.000%
1.2.83	276.00	+2.222%	+2.222%
1.3.83	269.00	-2.536%	-0.370%
1.4.83	343.00	+27.509%	+27.037%
1.5.83	345.00	+0.583%	+27.778%
1.6.83	333.00	-3.478%	+23.333%
1.7.83	338.00	+1.502%	+25.185%
1.8.83	367.00	+8.580%	+35.926%
1.9.83	390.00	+6.267%	+44.444%
1.10.83	378.00	-3.077%	+40.000%
1.11.83	384.00	+1.587%	+42.222%
1.12.83	404.00	+5.208%	+49.630%
1.1.84	415.00	+2.723%	+53.704%
1.2.84	425.00	+2.410%	+57.407%
1.3.84	399.00	-6.118%	+47.778%
1.4.84	410.00	+2.757%	+51.852%
1.5.84	388.00	-5.366%	+43.704%
12.5.84	376.00	-3.093%	+39.259%
1.6.84	397.00	+5.585%	+47.037%
1.7.84	430.00	+8.312%	+59.259%

Schneller Überblick über die Kursentwicklung



Die Kursentwicklung ist auch grafisch darstellbar

– Programm beenden
Verlassen Sie Shares in jedem Fall nach der Datensicherung über diesen Menüpunkt, denn er beendet die Programmausführung normal und löscht das Programm dabei nicht, so daß Sie bei einem versehentlichen Ende mit »GOTO 100« das Programm ohne Datenverlust neu starten.

(W. Meier/ja)

Steckbrief	
Programm:	Shares
Computer:	CPC 464/664/6128
Checksummer:	Explora
Datenträger:	Kassette, Diskette

```
10 REM "SHARES" - Programm zur Kursbeoba
   chtung von Wertpapieren                [BE24]
20 REM (c) Wolfgang Meier, Schinkelstr.2
   6, D-3340 Wolfenbuettel                [4928]
30 REM     Tel. 05331/72410               [6E46]
40 '                                      [8256]
50 INK 0,0:INK 1,24:PAPER 0:BORDER 0:PEN
    1:DIM a$(9,300,1):dz=0:ri=0:DIM S(1)
   :SYMBOL AFTER 64:ON ERROR GOTO 1310    [DF48]
60 SYMBOL 94,&X111100,&X1100110,&X110001
   10,&X1100i100,&X11001110,&X11000110,&
   X11011100,0:SYMBOL 64,0,&X1100110,0,&
   X1100110,&X1100110,&X1100110,&X111110
   ,0:SYMBOL 124,&X1100110,0,&X1100110,&
   X1100110,&X1100110,&X1100110,&X111100
   ,0                                     [0B62]
70 SYMBOL 91,&X11001100,0,&X1111000,&X11
   00,&X1111100,&X11001100,&X1110110:SYM
   BOL 123,&X1100110,&X11000,&X111100,&X
   1100110,&X1111110,&X1100110,&X1100110
   ,0:SYMBOL 93,&X1100110,0,&X111100,&X1
   100110,&X1100110,&X1100110,&X111100,0
                                          [20E6]
80 SYMBOL 125,&X11000110,&X111000,&X1101
   100,&X11000110,&X11000110,&X1101100,&
   X111000,0:SYMBOL 251,&X100100,&X10010
   ,&X1001,&X111,&X1001,&X10010,&X100100
   ,0:SYMBOL 252,&X10000000,&X1000000,&X
   100000,255,&X100000,&X1000000,128,0:S
   YMBOL 253,0,0,0,255,0,0,0,0            [F138]
90 SYMBOL 254,2,1,0,255,0,1,2,0:SYMBOL 2
   55,0,128,&X11100000,&X11111000,&X1110
   0000,128,0,0:p$=CHR$(251)+CHR$(252)+C
   HR$(253)+CHR$(254)+CHR$(255):pl=1:OPE
   NOUT"X":MEMORY HIMEM-1:CLOSEOUT        [26C6]
100 MODE 2                                [9C50]
110 LOCATE 29,1:PRINT"S H A R E S":LOCAT
    E 16,3:PRINT"Ein Programm zur Kursbe
    obachtung von Wertpapieren"           [99F0]
120 LOCATE 23,5:PRINT"Bitte w[hlen Sie:"
                                          [ACD0]
130 LOCATE 21,7:PRINT"Laden vorhandener
    Daten"                                [3A12]
140 LOCATE 21,9:PRINT"Dateieninhaltsverz
    eichnis"                              [846E]
150 LOCATE 21,11:PRINT"Neuanlage einer W
    ertpapierdatei"                       [A82E]
160 LOCATE 21,13:PRINT"Aktualisierung ei
    ner Wertpapierdatei"                  [4EA4]
170 LOCATE 21,15:PRINT"Kursdaten eines W
    ertpapiers"                           [1056]
180 LOCATE 21,17:PRINT"Kursentwicklung e
    ines Wertpapiers"                     [189A]
190 LOCATE 21,19:PRINT"Kursgrafik eines
    Wertpapiers"                          [D432]
200 LOCATE 21,21:PRINT"Datensicherung"    [7776]
210 LOCATE 21,23:PRINT"Programm beenden"
                                          [AA80]
220 LOCATE 11,pl*2+5:PRINT p$:SOUND 2,30
    ,30,7:WHILE (JOY(0) AND 16)<>16:FOR
    t=1 TO 80:NEXT t                      [FF60]
230 IF (JOY(0) AND 1)=1 AND pl>1 THEN LO
    CATE 11,pl*2+5:PRINT"<5>":pl=pl-1:LO
    CATE 11,pl*2+5:PRINT p$:SOUND 1,pl*2
    0+10,10,5 ELSE IF (JOY(0) AND 2)=2 A
    ND pl<9 THEN LOCATE 11,pl*2+5:PRINT"
    <5>":pl=pl+1:LOCATE 11,pl*2+5:PRINT
    p$:SOUND 1,pl*20+10,10,5              [553C]
240 WEND                                  [35C8]
250 SOUND 2,70,10,7:BORDER 26:FOR t=1 TO
     50:NEXT t:BORDER 0:ON pl GOSUB 270,
    290,330,500,760,890,1020,1170,1200    [0E68]
260 GOTO 100                              [D144]
270 CLS:ERASE A$:DZ=0:DIM A$(9,300,1):LO
    CATE 1,2:PRINT p$;"<4>Laden vorhande
    ner Daten":LOCATE 10,4:PRINT"Bitte s
    tellen Sie sicher, ob Ihre Datencass
    ette":LOCATE 10,5:PRINT"auf den rich
    tigen Z[hlerstand gebracht ist !":PR
    INT                                   [EA9E]
280 PRINT TAB(10);"Dr@cken Sie PLAY, dan
    n ENTER.":WHILE INKEY$<>CHR$(13):WEN
    D:OPENIN"!shares.dat":INPUT#9,DZ:FOR
     z1=0 TO dz-1:INPUT#9,A$(Z1,0,0):INP
    UT#9,A$(Z1,0,1):FOR Z2=1 TO VAL(A$(Z
    1,0,1))+2:INPUT#9,A$(Z1,Z2,0):INPUT#
    9,A$(Z1,Z2,1)                         [1262]
285 NEXT Z2,Z1:CLOSEIN:RETURN             [7D84]
290 CLS:GOSUB 1210:LOCATE 1,2:PRINT p$;"
    <4>Dateieninhaltsverzeichnis":PRINT:
    IF dz=0 THEN PRINT TAB(10);"Es sind
    keine Dateien vorhanden.":GOTO 320    [E626]
300 FOR z=0 TO dz-1:PRINT TAB(10);"Name:
     ";a$(z,0,0);TAB(46);"Gr]^e: ";USING
    "###";VAL(a$(z,0,1));:PRINT" Element
    ";:IF a$(z,0,1)<>"1"THEN PRINT"e"     [B806]
310 NEXT z                                [0C1A]
320 WHILE INKEY$="":WEND:RETURN           [EDEA]
330 CLS:GOSUB 1210:LOCATE 1,2:PRINT p$;"
    <4>Neuanlage einer Wertpapierdatei":
    IF dz<10 THEN 360 ELSE LOCATE 10,4:P
    RINT"Leider ist der Speicherplatz vo
    llst[ndig belegt,":PRINT TAB(10);"so
     da^ keine weiteren Dateien mehr ang
    elegt werden k]nnen."                 [645E]
340 PRINT:PRINT TAB(10);"Sollen daher di
    e vorhandenen Daten gel]scht werden
    ? ";:INPUT"",j$:IF LOWER$(j$)="ja"TH
    EN ERASE a$:DIM a$(9,300,1):dz=0      [D7E2]
350 IF INKEY$=""THEN 350 ELSE RETURN      [8E78]
360 LOCATE 10,4:PRINT"Bitte geben Sie de
    n Namen des neu zu registrierenden W
    ertpapiers ein.":PRINT:PRINT TAB(10)
    ;:INPUT"", a$(dz,0,0):IF LEN(a$(dz,0
    ,0))=0 THEN RETURN                    [1226]
370 IF LEN(a$(dz,0,0))>20 THEN a$(dz,0,0
    )=LEFT$(a$(dz,0,0),20)                [81AE]
380 PRINT:PRINT TAB(10);"Bitte geben Sie
     den Zeitpunkt des ersten zu registr
    ierenden Kurses ein."                 [5D06]
390 PRINT TAB(10);:INPUT"", a$(dz,3,0):I
    F LEN(a$(dz,3,0))=0 THEN PRINT CHR$(
    11);:GOTO 390                         [FA4A]
400 IF LEN(a$(dz,3,0))>15 THEN a$(dz,3,0
    )=LEFT$(a$(dz,3,0),15)                [4FC4]
410 a$(dz,1,0)=a$(dz,3,0):a$(dz,2,0)=a$(
    dz,3,0):PRINT:PRINT TAB(10);"Bitte g
    eben Sie den ersten zu registrierend
    en Kurs ein.":PRINT                   [5892]
420 PRINT TAB(10);SPC(70)                 [528A]
430 PRINT CHR$(11);TAB(10);:INPUT"", e$:
    IF LEN(e$)=0 OR e$="0" THEN PRINT CH
    R$(11);:GOTO 420                      [38B0]
440 z1=0:z2=0:FOR z=1 TO LEN(e$):IF MID$
    (e$,z,1)="." THEN z1=z1+1:GOTO 470    [EECE]
450 IF MID$(e$,z,1)>="0" AND MID$(e$,z,1
    )<="9" THEN z2=z2+1:GOTO 470          [3A48]
460 PRINT CHR$(11);:GOTO 420              [F4FE]
470 NEXT z                                [2128]
480 IF z1>1 OR z2<1 THEN PRINT CHR$(11);
    :GOTO 420                             [9B66]
490 a$(dz,1,1)=e$:a$(dz,2,1)=e$:a$(dz,3,
    1)=e$+"c":a$(dz,0,1)="1":dz=dz+1:RET
    URN                                   [4D48]
500 CLS:IF dz=0 THEN RETURN ELSE GOSUB 1
    210:WINDOW#0,1,80,4,25:PAPER#0,0:PEN
    #0,1:WINDOW#1,1,80,1,3:PAPER#1,0:PEN
    #1,1:LOCATE#1,2,2:PRINT#1,p$;"<3>Akt
    ualisierung einer Wertpapierdatei":M
    OVE 0,399:DRAWR 639,0:DRAWR 0,-47:DR
    AWR -639,0:DRAWR 0,47:GOSUB 1220      [3F3E]
510 GOSUB 1210:LOCATE#1,50,2:PAPER#1,1:P
    EN#1,0:PRINT#1,n$:PAPER#1,0:PEN#1,1   [2810]
520 IF ri=1 THEN ri=0:RETURN              [CA00]
530 IF VAL(a$(dn,0,1))<298 THEN 560       [C4CA]
540 PRINT TAB(10);"Die Kapazit[t dieser
    Datei ist bereits voll ausgenutzt,":
    PRINT TAB(10);"so da^ f@r dieses Wer
    tpapier keine Daten mehr eingegeben
    werden k]nnen."                       [3554]
550 IF INKEY$=""THEN 550 ELSE RETURN      [B880]
560 PRINT                                 [6390]
570 PRINT TAB(10);"Bitte geben Sie den Z
    eitpunkt des neuen Kurswerts ein.":P
    RINT                                  [ADE4]
580 PRINT TAB(10):INPUT"",f$              [C76E]
590 IF LEN(f$)=0 THEN PRINT CHR$(11);:GO
    TO 580                                [1A9E]
600 IF LEN(f$)>15 THEN f$=LEFT$(f$,15)    [7E6A]
610 a$(dn,VAL(a$(dn,0,1))+3,0)=F$:PRINT:
    PRINT TAB(10);"Bitte geben Sie den n
```

Die Kursentwicklung verschiedenster Wertpapiere haben Sie mit »Shares« im Griff

```
    euen Kurswert ein.":PRINT                [D7F6]
620 PRINT TAB(1Ø);SPC(7Ø):PRINT CHR$(11)
    ;TAB(1Ø);:INPUT"",e$:IF LEN(e$)=Ø OR
     e$="Ø" THEN PRINT CHR$(11);:GOTO 62
    Ø ELSE z1=Ø:z2=Ø:FOR z=1 TO LEN(e$):
    IF MID$(e$,z,1)="." THEN z1=z1+1:GOT
    O 65Ø                                    [6238]
63Ø IF MID$(e$,z,1)>="Ø" AND MID$(e$,z,1
    )<="9" THEN z2=z2+1:GOTO 65Ø             [D648]
64Ø PRINT CHR$(11);:GOTO 62Ø                 [F302]
65Ø NEXT z                                   [C328]
66Ø IF z1>1 OR z2<1 THEN PRINT CHR$(11);
    :GOTO 62Ø                                [A16A]
67Ø a$(dn,VAL(a$(dn,Ø,1))+3,1)=e$            [604E]
68Ø IF VAL(e$)>VAL(a$(dn,1,1)) THEN a$(d
    n,1,Ø)=f$:a$(dn,1,1)=e$:SOUND 1,3Ø,5
    Ø,7:PRINT:PRINT TAB(1Ø);"Neuer H]chs
    tkurs !" ELSE IF VAL(e$)<VAL(a$(dn,2
    ,1)) THEN a$(dn,2,Ø)=f$:a$(dn,2,1)=e
    $:SOUND 1,4ØØ,5Ø,7:PRINT:PRINT TAB(1
    Ø);"Neuer Tiefstkurs !"                  [93EA]
69Ø PRINT:PRINT TAB(1Ø);"Soll der neue K
    urs in die Kursstatistik einbezogen
    werden ?":PRINT:PRINT TAB(1Ø);           [E8D8]
7ØØ k$=LOWER$(INKEY$):IF k$="j"THEN PRIN
    T"ja":GOTO 71Ø ELSE IF k$="n" THEN P
    RINT"nein":GOTO 72Ø ELSE 7ØØ             [D794]
71Ø a$(dn,VAL(a$(dn,Ø,1))+3,1)=a$(dn,VAL
    (a$(dn,Ø,1))+3,1)+"c"                    [7E1E]
72Ø a$(dn,Ø,1)=STR$(VAL(a$(dn,Ø,1))+1):P
    RINT:PRINT"Wollen sie f@r dieses Wer
    tpapier weitere Daten eingeben, dr@c
    ken Sie ENTER,<5>andernfalls die Feu
    ertaste"                                 [D364]
73Ø IF INKEY$=CHR$(13) THEN CLS:GOTO 53Ø
                                             [1AF8]
74Ø IF (JOY(Ø) AND 16) <>16 THEN 73Ø         [A964]
75Ø RETURN                                   [BE38]
76Ø CLS:IF dz=Ø THEN RETURN ELSE GOSUB 1
    21Ø:WINDOW#Ø,1,8Ø,6,25:PAPER#Ø,Ø:PEN
    #Ø,1:WINDOW#1,1,8Ø,1,5:PAPER#1,Ø:PEN
    #1,1:LOCATE#1,2,2:PRINT#1,p$;"<3>Kur
    sdaten eines Wertpapiers":MOVE Ø,399
    :DRAWR 639,Ø:DRAWR Ø,-47:DRAWR -639,
    Ø:DRAWR Ø,47:GOSUB 122Ø                  [A802]
77Ø GOSUB 121Ø                               [A544]
78Ø IF ri=1 THEN ri=Ø:RETURN                 [1C10]
79Ø PAPER#1,1:PEN#1,Ø:LOCATE#1,4Ø,2:PRIN
    T#1,n$:PAPER#1,Ø:PEN#1,1:MOVE 48,341
    :TAG:PRINT"Datum";: MOVE 216,341:TAG
    :PRINT"Kurswert";:MOVE 4Ø8,341:TAG:P
    RINT"Kommentar";:TAGOFF:MOVE Ø,325:D
    RAW 639,325:lc=Ø:HIN=Ø:TIN=Ø:O=VAL(a
    $(dn,Ø,1))                               [D950]
8ØØ FOR z=1 TO VAL(a$(dn,Ø,1)):PRINT TAB
    (7);a$(dn,z+2,Ø);:PRINT TAB(28);USIN
    G"#####.##";VAL(a$(dn,z+2,1));           [4D6A]
81Ø IF VAL(a$(dn,z+2,1))=VAL(a$(dn,1,1))
     AND HIN=Ø THEN PRINT TAB(52);"H]chs
    tkurs";:HIN=1  ELSE IF VAL(a$(dn,z+2
    ,1))=VAL(a$(dn,2,1)) AND TIN=Ø THEN
    PRINT TAB(52);"Tiefstkurs";:TIN=1        [B07A]
82Ø lc=lc+1                                  [39E2]
83Ø IF lc=2Ø AND O<>z THEN lc=Ø:LOCATE#1
    ,7Ø,2:PRINT#1,"enter ?":SOUND 1,4Ø,3
    Ø,7:WHILE INKEY$<>CHR$(13):WEND:LOCA
    TE#1,7Ø,2:PRINT#1,"<7>"                  [D9BE]
84Ø IF (JOY(Ø) AND 16)=16 THEN 88Ø           [9AB8]
85Ø NEXT z                                   [E52C]
86Ø LOCATE#1,64,2:PRINT#1,"Ausgabeende":
    SOUND 1,2ØØ,3Ø,7                         [DCC2]
87Ø IF INKEY$=""THEN 87Ø                     [4C02]
88Ø RETURN                                   [C140]
89Ø CLS                                      [8846]
9ØØ IF dz=Ø THEN RETURN                      [0B04]
91Ø GOSUB 121Ø:WINDOW#Ø,1,8Ø,6,25:PAPER#
    Ø,Ø:PEN#Ø,1:WINDOW#1,1,8Ø,1,5:PAPER#
    1,Ø:PEN#1,1:LOCATE#1,2,2:PRINT#1,p$;
    "<3>Kursentwicklung eines Wertpapier
    s":MOVE Ø,399:DRAWR 639,Ø:DRAWR Ø,-4
    7:DRAWR -639,Ø:DRAWR Ø,47:GOSUB 122Ø
    :GOSUB 121Ø                              [59DA]
92Ø IF ri=1 THEN ri=Ø:RETURN                 [7608]
93Ø PAPER#1,1:PEN#1,Ø:LOCATE#1,45,2:PRIN
    T#1,n$:PAPER#1,Ø:PEN#1,1:MOVE 8,341:
    TAG:PRINT"Datum";:MOVE 176,341:TAG:P
    RINT"Kurswert";:MOVE 28Ø,341:TAG:PRI
    NT"Ver[nderung";:MOVE 416,341:TAG:PR
    INT"Gesamtver[nderung";:TAGOFF:MOVE
    Ø,325:DRAW 639,325                       [3508]
94Ø O=VAL(a$(dn,Ø,1)):lc=Ø:x$="":FOR z=1
     TO O:IF RIGHT$(A$(dn,z+2,1),1)<>"c"
     THEN 98Ø                                [E4B0]
95Ø y$=a$(dn,z+2,1):PRINT TAB(2);a$(dn,z
    +2,Ø);:PRINT TAB(23);USING"#####.##"
    ;VAL(y$);:IF x$<>""THEN PRINT TAB(36
    );USING"+####.###";(VAL(y$)/VAL(x$)-
    1)*1ØØ;:PRINT"%";                        [90FE]
96Ø PRINT TAB(53);USING"+#####.###";(VAL
    (y$)/VAL(a$(dn,3,1))-1)*1ØØ;:PRINT"%
    ";:x$=y$:lc=lc+1:IF lc=2Ø AND O<>z T
    HEN lc=Ø:LOCATE#1,7Ø,2:PRINT#1,"ente
    r ?":SOUND 1,4Ø,3Ø,7:WHILE INKEY$<>C
    HR$(13):WEND:LOCATE#1,7Ø,2:PRINT#1,"
    <7>"                                     [14A8]
97Ø IF (JOY(Ø) AND 16)=16 THEN 1Ø1Ø          [D604]
98Ø NEXT z                                   [2434]
99Ø LOCATE#1,67,2:PRINT#1,"Ausgabeende":
    SOUND 1,2ØØ,3Ø,7                         [98D0]
1ØØØ IF INKEY$="" THEN 1ØØØ                  [72CA]
1Ø1Ø RETURN                                  [8784]
1Ø2Ø CLS                                     [0F8A]
1Ø3Ø IF dz=Ø THEN RETURN                     [8C5A]
1Ø4Ø GOSUB 121Ø:PRINT p$;"<4>Kursgrafik
     eines Wertpapiers":GOSUB 122Ø:GOSUB
      121Ø:IF RI=1 THEN RI=Ø:RETURN          [890C]
1Ø5Ø CLS:PRINT P$;"<4>Kursgrafik eines W
     ertpapiers":PAPER 1:PEN Ø:LOCATE 5Ø
     ,1:PRINT A$(dn,Ø,Ø):PAPER Ø:PEN 1:O
     RIGIN 3,4Ø:MOVE -1,-2:DRAWR 635,Ø,1
     :DRAWR Ø,338:DRAWR -635,Ø:DRAWR Ø,-
     338:ec=VAL(a$(dn,Ø,1)):tk=VAL(a$(dn
     ,2,1)):hk=VAL(a$(dn,1,1))               [E00E]
1Ø6Ø ERASE s:DIM s(ec):FOR z=1 TO ec:s(z
     )=VAL(a$(dn,2+z,1)):NEXT z:IF EC>1
     THEN xd=633/(EC-1)                      [DDD8]
1Ø7Ø yd=(hk-tk)/335                          [65F8]
1Ø8Ø IF yd=Ø THEN yd=1:TK=S(1)-166           [83A2]
1Ø9Ø PLOT Ø,ROUND((s(1)-tk)/yd):FOR z=2
     TO ec:DRAW ROUND((z-1)*xd),ROUND((s
     (z)-tk)/yd):NEXT z:LOCATE 1,24:PRIN
     T"Zeitpunkt: ":LOCATE 41,24:PRINT"K
     urs: ":PRINT CHR$(23);CHR$(1);:sx=1
     :PAPER 1:PEN Ø:GOTO 111Ø                [9F16]
11ØØ IF (JOY(Ø) AND 4)=4 AND sx>1 THEN G
     OSUB 116Ø:sx=sx-1 ELSE IF (JOY(Ø) A
     ND 8)=8 AND sx<ec THEN GOSUB 116Ø:s
     x=sx+1 ELSE 115Ø                        [F894]
111Ø GOSUB 116Ø                              [4296]
112Ø SOUND 1,4Ø,1Ø,6                         [927A]
113Ø LOCATE 12,24:PRINT" ";USING"\<15>\"
     ;a$(dn,sx+2,Ø)                          [235A]
114Ø LOCATE 47,24:PRINT" ";USING"#####.#
     #";s(sx)                                [61F0]
115Ø IF (JOY(Ø) AND 16)<>16 THEN 11ØØ EL
     SE PRINT CHR$(23);CHR$(Ø);:PAPER Ø:
     PEN 1:RETURN                            [C86A]
116Ø gx= ROUND((sx-1)*xd)-1:gy=ROUND((s(
     sx)-tk)/yd)+6:PLOT gx,gy:PLOTR 1,Ø:
     PLOTR 1,Ø:PLOTR 1,-2:PLOTR 1,-2:PLO
     TR Ø,-2:PLOTR Ø,-2:PLOTR -1,-2:PLOT
     R -1,-2:PLOTR -1,Ø:PLOTR -1,Ø:PLOTR
      -1,2:PLOTR -1,2:PLOTR Ø,2:PLOTR Ø,
     2:PLOTR 1,2:RETURN                      [6CF0]
117Ø CLS:LOCATE 1,2:PRINT p$;"<4>Datensi
     cherung":LOCATE 1Ø,4:PRINT"Bitte st
     ellen Sie sicher, ob Ihre Datencass
     ette":LOCATE 1Ø,5:PRINT"auf den ric
     htigen Z[hlerstand gebracht ist !"      [8A1A]
118Ø PRINT:PRINT TAB(1Ø);"Dr@cken Sie RE
     C und PLAY, dann ENTER."                [CA5C]
119Ø WHILE INKEY$<>CHR$(13):WEND:SPEED W
     RITE 1:OPENOUT"!shares.dat":PRINT#9
     ,dz:FOR z1=Ø TO dz-1:PRINT#9,a$(z1,
     Ø,Ø):PRINT#9,a$(z1,Ø,1):FOR z2=1 TO
      VAL(a$(z1,Ø,1))+2:PRINT#9,a$(z1,z2
     ,Ø):PRINT#9,a$(z1,z2,1):NEXT z2,z1:
     CLOSEOUT:RETURN                         [1286]
12ØØ CLS:GOSUB 121Ø:LOCATE 1,2:PRINT p$;
     "<4>Programm beenden":PRINT:PRINT:P
     RINT:END                                [0934]
121Ø IF INKEY$<>""THEN 121Ø ELSE RETURN      [1DA2]
122Ø PRINT:PRINT TAB(1Ø);"Bitte w[hlen S
     ie das gew@nschte Wertpapier:":PRIN
     T:pe=1:PRINT TAB(4);:PRINT TAB(11);
     p$;:j=VPOS(#Ø)-1:FOR z=Ø TO dz-1:PR
     INT TAB(21);a$(z,Ø,Ø):NEXT z:PRINT
     TAB(21);"HAUPTMEN!"                     [65EC]
123Ø WHILE (JOY(Ø) AND 16)<>16               [633C]
124Ø IF (JOY(Ø) AND 1)=1 AND pe>1 THEN L
     OCATE 11,pe+j:PRINT"<5>":pe=pe-1:LO
     CATE 11,pe+j:PRINT p$:SOUND 1,pe*2Ø
     +3Ø,1Ø,5                                [04DC]
125Ø IF (JOY(Ø) AND 2)=2 AND pe<dz+1 THE
     N LOCATE 11,pe+j:PRINT"<5>":pe=pe+1
     :LOCATE 11,pe+j:PRINT p$:SOUND 1,pe
     *2Ø+3Ø,1Ø,5                             [EEEC]
126Ø FOR t=1 TO 5Ø:NEXT t                    [DE4E]
127Ø WEND                                    [1830]
128Ø IF pe=dz+1 THEN ri=1:GOTO 13ØØ          [B0EA]
129Ø dn=pe-1:n$=a$(dn,Ø,Ø)                   [B12E]
13ØØ CLS:RETURN                              [3EC0]
131Ø CLS                                     [1E8E]
132Ø ON ERROR GOTO 131Ø                      [3FD6]
133Ø GOSUB 121Ø:SOUND 1,5Ø,2Ø,6:MODE Ø:L
     OCATE 3,12:PRINT"Speicher voll !":F
     OR w=1 TO 15ØØ:NEXT w                   [DC3C]
134Ø IF INKEY$="" THEN 134Ø ELSE RESUME
     1ØØ                                     [50BC]
```

»Shares« verwaltet Ihr Vermögen (Schluß)

Datenexpreß

Eine Speichererweiterung als RAM-Floppy ist schon etwas Feines. Ohne geeignetes Programm nutzt sie jedoch herzlich wenig.

Eine Speichererweiterung ist Retter in der Speicherplatz-Not, sofern sie über ein intelligentes Betriebssystem verfügt. Aber auch dann gilt der Grundsatz: Ohne Software nutzt die leistungsfähigste Hardware rein gar nichts. Aus diesen Überlegungen heraus entstand das Programm »RAMDISK« für die Vortex-Speichererweiterungen.

Eigentlich besteht »RAMDISK« aus einem ganzen Programmpaket: »RAMDISK.BAS« (Listing 1) aktiviert das Bank-Basic und enthält auch die Dokumentation. (Diesen Teil dürfen Sie weglassen, wenn Sie vor dem Laden des zweiten Teils mit »IBOS« manuell umschalten.) »RAMDISK.BOS« (Listing 2) übernimmt die Definition des deutschen Zeichensatzes, Tastaturbelegung und Farbzuordnung. »RAMDISK.RAM« besteht aus den beiden Teilen »RAMDISK.BK1« (Listing 3) und »RAMDISK.BK0« (Listing 4). Diese zwei letzten Listings geben Sie zusammen ein und speichern sie mit »A$= "RAMDISK.RAM": SAVE, @ A$«.

Bei der Arbeit mit der Datei müssen Sie beachten, daß Sie immer das erste Datenfeld jedes Datensatzes beschreiben. Wenn Sie später einzelne Datensätze aus der Datei löschen wollen, dürfen Sie keinesfalls versehentlich diese Löschung doppelt vornehmen. Bei der Suchfunktion ist es nicht nötig, den Suchbegriff exakt vorzugeben. Mit der Eingabe »Dampf« finden Sie so beispielsweise nicht nur die Dampflok, sondern auch die Dampfmaschine. Auch statistische Auswertungen sind machbar, da das Programm Summen berechnet. Nach jedem Speicher-Vorgang wird automatisch ein »CAT« ausgeführt, so läßt sich bei Verwendung eines Kassettenrecorders als Speichermedium die korrekte Datenaufzeichnung überprüfen.

Noch ein kleiner Hinweis zur Eingabe der beiden »RAMDISK.RAM«-Teile: »RAMDISK.BK1« kommt in die Bank 1 (vorher mit »IBANK,1« aktivieren) und »RAMDISK.BK0« in Bank 0. Beim Speichern mit der oben erwähnten Befehlsfolge entstehen dann daraus die Dateien »RAMDISK.RAM«, »RAMDISK.BK1« und »RAMDISK.BK0«.

(Karlheinz Battermann/ja)

Steckbrief	
Programm:	RAMDISK
Computer:	CPC 464/664
Checksummer:	Explora
Datenträger:	Kassette, Diskette
Besonderes:	nur mit Vortex-Speichererweiterung

```
10  '************************************
                                          [020C]
20  '***       MULTI - RAM DISK        ***
                                          [4810]
30  '***                               ***
                                          [E7CC]
40  '***  Fuer SCHNEIDER CPC mit       ***
                                          [159C]
50  '***  VORTEX RAM-Erweiterungen     ***
                                          [E3C2]
60  '***  SP64/M und SP64-512          ***
                                          [2CD6]
70  '************************************
                                          [C018]
80  '*** (c) Karlheinz Battermann      ***
                                          [6326]
90  '***     Weissdornweg 20           ***
                                          [F51E]
100 '***     3422 Bad Lauterberg - 1 ***
                                          [77A2]
110 '***     Tel. 05524/2530           ***
                                          [4602]
120 '************************************
                                          [F230]
130 '                                     [DFB6]
140 '************************************
                                          [FA34]
150 '***          RAMDISK.BAS          ***
                                          [9E50]
160 '************************************
                                          [AA38]
170 '                                     [03BE]
180 '***  Bank-BASIC  BOS 1.0 ein-     ***
                                          [2AC6]
190 '***  schalten ....                ***
                                          [670E]
200 '                                     [E4B2]
210 |BOS                                  [F226]
220 '                                     [06B6]
230 '************************************
    ************************************
    ***                                   [5900]
240 '***     Programm-Dokumentation
      ***         Variablen-Tabelle
    ***                                   [B2CA]
250 '************************************
    ************************************
    ***                                   [3304]
260 '*** Variable    * Benutzung in Prog
    ramm RAMDISK.RAM fuer ....
    ***                                   [8C58]
270 '************************************
    ************************************
    ***                                   [AD08]
280 ' c                Die Zeilen 130/14
    0 duerfen nur einmal durchlaufen wer
    den                                   [5EBE]
290 ' c$               COMMON-Variablen-
    Definition                            [3C52]
300 ' t                Zeitschleife       [8D9A]
310 ' t$               Tastatur (INKEY)
    - Abfrage                             [4EA2]
320 ' e$,f$,g$,h$,i$   Text fuer Window
    2                                     [B448]
330 ' p,p1             Eingabe von Progr
    amm-, Dateikarten-,Spalten Nr.        [2E00]
340 ' rd               Ram Disk noch fre
    i?  nein/ja                           [5060]
350 ' n$               Dateiname          [D1F2]
360 ' da$              Dateidatum         [3256]
370 ' s                Anzahl der Dateis
    palten                                [5E56]
380 ' ss               Anzahl der Dateis
    palten (Schutzvariable)               [7F6E]
390 ' b1 - b15         Anzahl Buchstaben
     pro Dateispalte                      [4DD2]
400 ' x                Eingabe Anzahl Bu
    chstaben pro Spalte ueber Monitor/Ca
    ss.                                   [53AE]
410 ' r                Buchstabensumme a
    ller Dateispalten fuer :ramopen-Befe
    hl                                    [19A6]
420 ' dk               Anzahl der moegli
    chen Dateikarten (rec-1)              [3D6A]
430 ' z                Dateikarten Nr.    [52E4]
440 ' z1               hoechste eingegeb
    ene Dateikarten Nr.                   [3F0A]
450 ' s1               Hilfsvariable fue
    r Dateikartenbearbeitung              [6B4C]
460 ' a                Anfangsnummer (Ei
    ngabe in Dateikarten,Suchen,Drucken)
                                          [33BA]
470 ' e                Endnummer     (Ei
    ngabe in Dateikarten,Suchen,Drucken)
                                          [3D76]
480 ' d$               Dateneingabe       [2244]
490 ' bb               Kontrolle der Dat
    eneingabelaenge (Buchstaben pro Spal
    te)                                   [44D4]
500 ' d1$ - d15$       Daten der einzeln
    en Dateispalten                       [8C3A]
510 ' d                Hilfsvariable im
    WINDOW 2 - INPUT (Zeilenansteuerung)
                                          [03C6]
520 ' ds               Dateisicherung er
    forderlich?  nein/ja                  [6440]
530 ' b$(s)            Feld fuer Bezeich
    nungen                                [D3F6]
540 ' b                Bezeichnungen ein
    gegeben?  nein/ja                     [7FE6]
550 ' dl(10,2)         Feld fuer geloesc
    hte Dateikarten                       [893A]
560 ' lz               Zaehler fuer Feld
     dl                                   [32E0]
570 ' z2               Zaehler beim Such
```

```
     en geloeschter Dateikarten                          [62BC]
580  ' si$            Mass-Einheit der
     Inventarsummen                                      [9D44]
590  ' k$,k1$,k       Inventareinheit (
     DM,kg,m usw)                                        [C6F8]
600  ' st$,st1$,st    Stueckzahl                         [C3C2]
610  ' si             Inventarsummen we
     lcher Dateispalte ermitteln                         [1D48]
620  ' sz             Stueckzahlen welc
     her Dateispalte ermitteln                           [3996]
630  ' su             Inventarsumme (k*
     st)                                                 [1D46]
640  ' sw,sw1,sw2     Laenge des Suchwo
     rtes                                                [D6B2]
650  ' sw$,sw1$,sw2$  Name des Suchwort
     es                                                  [8536]
660  ' swa            Suchwort gefunden
     ? nein/ja                                           [28F8]
670  ' swd            Suchwortdurchlauf
     bei 2 Suchwoertern                                  [A4F4]
680  ' zz             Zeilenzaehler bei
     m Dateiausdruck                                     [8FA2]
690  ' s2             Hilfsvariable bei
     m Ausdruck einer Datei mit 6-10 Spal
     ten                                                 [F0CA]
700  ' dz             Leerzeile nach Da
     teikartenausdruck?  nein/ja                         [9150]
710  '************************************
     ************************************
     ***                                                 [7D06]
```

Listing 1. Information für Wissensdurstige

```
10   '************************************              [020C]
20   '***       MULTI - RAM DISK         ***             [4810]
30   '***                                ***             [E7CC]
40   '***  Fuer SCHNEIDER CPC mit        ***             [159C]
50   '***  VORTEX RAM-Erweiterungen      ***             [E3C2]
60   '***  SP64/M und SP64-512           ***             [2CD6]
70   '************************************              [C018]
80   '*** (c) Karlheinz Battermann       ***             [6326]
90   '***      Weissdornweg 20           ***             [F51E]
100  '***      3422 Bad Lauterberg - 1 ***               [77A2]
110  '***      Tel. 05524/2530           ***             [4602]
120  '************************************              [F230]
130  '                                                   [DFB6]
140  '************************************              [FA34]
150  '***          RAMDISK.BOS           ***             [946C]
160  '************************************              [AA38]
170  '                                                   [03BE]
180  '***  deutsche Zeichen  ***                         [B32E]
190  '                                                   [05C2]
200  SYMBOL 91,198,16,56,108,198,254,198,
     0                                                   [A11C]
210  SYMBOL 92,102,56,108,198,198,108,56,
     0                                                   [7606]
220  SYMBOL 93,102,0,102,102,102,102,60,0                [1E36]
230  SYMBOL 123,102,0,120,12,124,204,118,
     0                                                   [03A2]
240  SYMBOL 124,102,0,60,102,102,102,60,0                [E536]
250  SYMBOL 125,102,0,102,102,102,102,59,
     0                                                   [A9A4]
260  SYMBOL 126,120,198,198,252,198,198,2
     48,192                                              [442A]
270  KEY DEF 17,1,123,91                                 [FB9C]
280  KEY DEF 19,1,124,92                                 [0BA6]
290  KEY DEF 22,1,125,93                                 [4CA0]
300  KEY DEF 26,1,64,126                                 [EE96]
310  '                                                   [E1B6]
320  '***  Die "TAB"-Taste zusammen  ***                 [6EC6]
330  '***  mit SHIFT gedrueckt er-   ***                 [D5C6]
340  '***  gibt den Befehl zum       ***                 [ADB2]
350  '***  "LISTEN" des Programmes!  ***                 [F450]
360  '                                                   [04C0]
370  KEY 141,"MODE 2:PEN 1:PAPER 0:CLS:LI
     ST"+CHR$(13)                                        [5E46]
380  '                                                   [0AC4]
390  '***  Die "TAB"-Taste zusammen  ***                 [2DD4]
400  '***  mit CTRL gedrueckt        ***                 [5928]
410  '***  fuehrt nach einem BREAK   ***                 [ABE6]
420  '***  usw. ohne Datenverlust    ***                 [EB22]
430  '***  zum Hauptmenue zurueck!   ***                 [29B8]
440  '                                                   [E4BE]
450  KEY 142,"GOTO 10"+CHR$(13)                          [B6C8]
460  KEY DEF 68,0,9,141,142                              [4ECC]
470  '                                                   [E3C4]
480  '***  Initialisierung  ***                          [471E]
490  '                                                   [B1C8]
500  MODE 2:BORDER 0:INK 0,1:INK 1,15                    [4D78]
510  WINDOW 1,80,1,3:PEN 0:PAPER 1:CLS:LO
     CATE 22,2:PRINT"* *{2 SPACE}M U L T
     I{2 SPACE}-{2 SPACE}R A M{3 SPACE}D
     I S K{2 SPACE}* *"                                  [F38A]
520  WINDOW 1,80,23,25:CLS:PRINT:PRINT"Pr
     ogramm{2 SPACE}* RAMDISK.RAM *{2 SPA
     CE}wird geladen!{2 SPACE}Bitte warte
     n ...."                                             [E400]
530  WINDOW 1,80,4,22:PEN 1:PAPER 0:CLS:L
     OCATE 1,4:PRINT STRING$(80,"-"):LOCA
     TE 1,14:PRINT STRING$(80,"-")                       [D9B0]
540  LOCATE 19,7:PRINT"Ein universelles D
     atei-Programm mit RAM DISK"                         [77E8]
550  LOCATE 8,9:PRINT"f}r SCHNEIDER CPC m
     it VORTEX RAM-Erweiterungen SP64/M u
     nd SP64-512"                                        [6F7E]
560  LOCATE 26,13:PRINT"(c) 1986 by kabas
     oft-computing"                                      [B656]
570  '                                                   [E3C6]
580  '***  Programm > RAMDISK.RAM <  ***                 [7430]
590  '***  nachladen ....            ***                 [3C2E]
600  '***  Programm > RAMDISK.BOS <  ***                 [402A]
610  '***  wird hierbei geloescht!   ***                 [B74E]
620  '                                                   [0EBE]
630  WINDOW 27,60,20,20:CLS:p$="RAMDISK.R
     AM":|RUN,@p$                                        [59EE]
```

Listing 2. Nun »spricht« Ihr CPC deutsch

```
10   '************************************              [020C]
20   '***       MULTI - RAM DISK         ***             [4810]
30   '************************************              [8410]
40   '                                                   [C896]
50   '************************************              [4E14]
60   '***          RAMDISK.RAM           ***             [5844]
70   '***            Bank 1              ***             [88EE]
80   '************************************              [F11A]
90   '                                                   [CEA0]
100  '***  Initialisierung  ***                          [4408]
110  '                                                   [DDB2]
120  e$="Programmziffer oder >ENTER< dr}c
     ken! ":h$="{2 SPACE}--> Taste >J< dr
     }cken!":i$=" Bitte warten ...."                     [2202]
130  d1$="":d2$="":d3$="":d4$="":d5$="":d
     6$="":d7$="":d8$="":d9$="":d10$="":d
     11$="":d12$="":d13$="":d14$="":d15$=
     ""                                                  [8032]
140  |RAMWRITE,0,@d1$,@d2$,@d3$,@d4$,@d5$
     ,@d6$,@d7$,@d8$,@d9$,@d10$,@d11$,@d1
     2$,@d13$,@d14$,@d15$                                [74EC]
150  |RETURN                                            [6724]
160  '                                                   [04BC]
170  '***  RAMREAD  Daten lesen  ***                     [8774]
180  '                                                   [06C0]
190  ON s-4 GOTO 200,210,220,230,240,250,
     260,270,280,290,300                                 [5BE0]
200  |RAMREAD,z,@d1$,@d2$,@d3$,@d4$,@d5$:
     RETURN                                              [1D5A]
210  |RAMREAD,z,@d1$,@d2$,@d3$,@d4$,@d5$,
     @d6$:RETURN                                         [17B0]
220  |RAMREAD,z,@d1$,@d2$,@d3$,@d4$,@d5$,
     @d6$,@d7$:RETURN                                    [E908]
230  |RAMREAD,z,@d1$,@d2$,@d3$,@d4$,@d5$,
     @d6$,@d7$,@d8$:RETURN                               [1E62]
240  |RAMREAD,z,@d1$,@d2$,@d3$,@d4$,@d5$,
     @d6$,@d7$,@d8$,@d9$:RETURN                          [8CBE]
250  |RAMREAD,z,@d1$,@d2$,@d3$,@d4$,@d5$,
     @d6$,@d7$,@d8$,@d9$,@d10$:RETURN                    [9D6A]
260  |RAMREAD,z,@d1$,@d2$,@d3$,@d4$,@d5$,
```

Listing 3. Inhalt der Bank 1

```
     @d6$,@d7$,@d8$,@d9$,@d10$,@d11$:RETU
     RN                                      [2818]
270  |RAMREAD,z,@d1$,@d2$,@d3$,@d4$,@d5$,
     @d6$,@d7$,@d8$,@d9$,@d10$,@d11$,@d12
     $:RETURN                                [78C8]
280  |RAMREAD,z,@d1$,@d2$,@d3$,@d4$,@d5$,
     @d6$,@d7$,@d8$,@d9$,@d10$,@d11$,@d12
     $,@d13$:RETURN                          [637A]
290  |RAMREAD,z,@d1$,@d2$,@d3$,@d4$,@d5$,
     @d6$,@d7$,@d8$,@d9$,@d10$,@d11$,@d12
     $,@d13$,@d14$:RETURN                    [B42E]
300  |RAMREAD,z,@d1$,@d2$,@d3$,@d4$,@d5$,
     @d6$,@d7$,@d8$,@d9$,@d10$,@d11$,@d12
     $,@d13$,@d14$,@d15$:RETURN              [13D2]
310  '                                       [E1B6]
320  '***  WINDOW 2  ***                     [F304]
330  '                                       [DFBA]
340  WINDOW 1,80,23,25:CALL &BB9C:CLS        [73A0]
350  PRINT:PRINT f$                          [1C6C]
360  WINDOW 1,80,4,22:CALL &BB9C             [FF04]
370  RETURN                                  [A634]
380  '                                       [0AC4]
390  '***  WINDOW 2 - INPUT  ***             [BC0C]
400  '                                       [08B6]
410  WINDOW 1,80,23,25:CALL &BB9C:CLS        [769C]
420  PRINT:PRINT f$+" ";                     [CCFC]
430  IF d=1 THEN LINE INPUT d$:IF LEN(d$)
     >20 THEN 410 ELSE 450                   [531C]
440  INPUT"",p                               [C650]
450  WINDOW 1,80,4,22:CALL &BB9C:d=0         [911A]
460  RETURN                                  [A934]
470  '                                       [E3C4]
480  '***  INKEY - ABFRAGE  ***              [B068]
490  '                                       [B1C8]
500  t$=INKEY$:IF t$="" THEN 500             [2768]
510  t$=UPPER$(t$)                           [B148]
520  RETURN                                  [AA2E]
530  '                                       [07BE]
540  '***  nach Suchwort suchen  ***         [5336]
550  '                                       [E1C2]
560  IF sw$=LEFT$(d1$,sw) THEN 750           [FE5E]
570  IF sw$=LEFT$(d2$,sw) THEN 750           [5B62]
580  IF sw$=LEFT$(d3$,sw) THEN 750           [8C66]
590  IF sw$=LEFT$(d4$,sw) THEN 750           [196A]
600  IF sw$=LEFT$(d5$,sw) THEN 750           [205C]
610  IF sw$=LEFT$(d6$,sw) THEN 750           [C360]
620  IF s<7 THEN 740                         [01AE]
630  IF sw$=LEFT$(d7$,sw) THEN 750           [A566]
640  IF sw$=LEFT$(d8$,sw) THEN 750           [1C6A]
650  IF sw$=LEFT$(d9$,sw) THEN 750           [6B6E]
660  IF s<10 THEN 740                        [EE0A]
670  IF sw$=LEFT$(d10$,sw) THEN 750          [2FC2]
680  IF sw$=LEFT$(d11$,sw) THEN 750          [57C6]
690  IF sw$=LEFT$(d12$,sw) THEN 750          [F7CA]
700  IF s<13 THEN 740                        [7A06]
710  IF sw$=LEFT$(d13$,sw) THEN 750          [B6BE]
720  IF sw$=LEFT$(d14$,sw) THEN 750          [D2C2]
730  IF sw$=LEFT$(d15$,sw) THEN 750          [EEC6]
740  swa=0:swd=0:RETURN                      [5F04]
750  swa=1:IF p1<>2 THEN RETURN              [0DF4]
760  IF swd=0 THEN sw=sw2:sw$=sw2$:swd=1:
     GOTO 560                                [4784]
770  swd=0:RETURN                            [C026]
780  '                                       [B2CC]
790  '***  Inventarkosten addieren  ***      [A134]
800  '                                       [10BE]
810  IF k1$="*" THEN k$="1":GOTO 980         [2D6E]
820  ON si GOTO 830,840,850,860,870,880,8
     90,900,910,920,930,940,950,960,970      [6FD4]
830  k$=d1$:GOTO 980                         [4AE8]
840  k$=d2$:GOTO 980                         [55EC]
850  k$=d3$:GOTO 980                         [0CF0]
860  k$=d4$:GOTO 980                         [37F4]
870  k$=d5$:GOTO 980                         [16F8]
880  k$=d6$:GOTO 980                         [11FC]
890  k$=d7$:GOTO 980                         [F900]
900  k$=d8$:GOTO 980                         [33F2]
910  k$=d9$:GOTO 980                         [04F6]
920  k$=d10$:GOTO 980                        [BF48]
930  k$=d11$:GOTO 980                        [BD4C]
940  k$=d12$:GOTO 980                        [5350]
950  k$=d13$:GOTO 980                        [C154]
960  k$=d14$:GOTO 980                        [7758]
970  k$=d15$                                 [F8F4]
980  IF st1$="*" THEN st$="1":GOTO 1150      [FEBA]
990  ON sz GOTO 1000,1010,1020,1030,1040,
     1050,1060,1070,1080,1090,1100,1110,1
     120,1130,1140                           [30B0]
1000 st$=d1$:GOTO 1150                       [1D78]
1010 st$=d2$:GOTO 1150                       [717C]
1020 st$=d3$:GOTO 1150                       [2580]
1030 st$=d4$:GOTO 1150                       [FD84]
1040 st$=d5$:GOTO 1150                       [5D88]
1050 st$=d6$:GOTO 1150                       [918C]
1060 st$=d7$:GOTO 1150                       [2D90]
1070 st$=d8$:GOTO 1150                       [FD94]
1080 st$=d9$:GOTO 1150                       [3D98]
1090 st$=d10$:GOTO 1150                      [51EA]
1100 st$=d11$:GOTO 1150                      [47DC]
1110 st$=d12$:GOTO 1150                      [36E0]
1120 st$=d13$:GOTO 1150                      [55E4]
1130 st$=d14$:GOTO 1150                      [3CE8]
1140 st$=d15$                                [4238]
1150 k=VAL(k$):st=VAL(st$):su=su+k*st        [81BA]
1160 RETURN                                  [C690]
1170 '                                       [B420]
1180 '***  Datei-Inventarsummen ermittel
     n  ***                                  [AAE2]
1190 '                                       [B624]
1200 CLS:ZONE 23                             [F780]
1210 PRINT:PRINT,"Berechnung von Inventa
     rsummen":PRINT,STRING$(29,"-")          [5CBE]
1220 PRINT:PRINT,"Es k!nnen die Inventar
     summen, die"                            [9DC6]
1230 PRINT:PRINT,"in jeder beliebigen Da
     teispalte"                              [90C2]
1240 PRINT:PRINT,"stehen d}rfen, berechn
     et werden!"                             [0840]
1250 LOCATE 1,11:PRINT,">1<{2 SPACE}nach
      1 Suchwort":PRINT:PRINT,">2<{2 SPA
     CE}nach 2 Suchw!rtern":PRINT:PRINT,
     ">3<{2 SPACE}der Gesamtdatei"           [6FC0]
1260 f$=e$:GOSUB 390:p1=p:IF p1<1 THEN |
     RETURN ELSE IF p1>3 THEN 1260           [23B6]
1270 f$="Inventarsummen welcher Spalte e
     rmitteln?{2 SPACE}Nr.":GOSUB 390:si
     =p:IF si<1 THEN 1260 ELSE IF si>15
     THEN 1270                               [CA72]
1280 f$="Inventarsummen der Spalte"+STR$
     (si)+" sind in welcher Ma~einheit (
     DM,kg,cm) angeben?":d=1:GOSUB 390:s
     i$=d$                                   [0ABE]
1290 f$="Gibt es zugeh!rige St}ckzahlen?
     "+h$:GOSUB 320:GOSUB 480:IF t$<>"J"
     THEN st1$="*":GOTO 1310                 [7942]
1300 f$="Die zugeh!rigen St}ckzahlen ste
     hen in welcher Spalte?{2 SPACE}Nr."
     :GOSUB 390:sz=p:IF sz<1 THEN st1$="
     *" ELSE IF sz>s THEN 1300               [69E6]
1310 IF p1=3 THEN swa=1:GOTO 1370            [6C56]
1320 f$="Bitte Suchwort eingeben! ":d=1:
     GOSUB 390:sw1=LEN(d$):sw1$=d$:IF sw
     1$="" THEN st1$="":GOTO 1200            [A776]
1330 IF p1=2 THEN f$="Bitte 2.Suchwort e
     ingeben! ":d=1:GOSUB 390:sw2=LEN(d$
     ):sw2$=d$:IF sw2$="" THEN 1330 ELSE
      1350                                   [8966]
1340 f$="Suchwort{2 SPACE}* "+sw1$+" *{2
      SPACE}-{2 SPACE}Summen werden ermi
     ttelt!"+i$:GOSUB 320:GOTO 1380          [CDCE]
1350 g$="Suchw!rter{2 SPACE}* "+sw1$+" *
     {2 SPACE}und{2 SPACE}* "+sw2$+" *":
     IF sw1+sw2 >23 THEN 1360 ELSE f$=g$
     +"{2 SPACE}-{2 SPACE}Summen werden
     ermittelt!"+CHR$(10)+CHR$(13)+"Bitt
     e warten ....":GOSUB 320:GOTO 1380      [E7D4]
1360 f$=g$+CHR$(10)+CHR$(13)+"Summen wer
     den ermittelt!"+i$:GOSUB 320:GOTO 1
     380                                     [754A]
1370 f$="Inventarsummen werden ermittelt
     !"+i$:GOSUB 320                         [7F72]
1380 FOR z=1 TO z1:GOSUB 170:IF p1<3 THE
     N sw=sw1:sw$=sw1$:GOSUB 540             [7D78]
1390 IF swa=0 THEN 1400 ELSE GOSUB 790       [0E62]
1400 NEXT:LOCATE 1,10:PRINT CHR$(20);CHR
     $(7)                                    [EEA6]
1410 PRINT:PRINT,"Die Inventarsummen der
      Spalte";si:IF p1=1 THEN PRINT:PRIN
     T,"zum Suchwort{2 SPACE}* ";sw1$;"
     *"                                      [730C]
1420 IF p1=2 THEN PRINT:PRINT,"zu den Su
     chw!rtern":PRINT:PRINT,"* ";sw1$;"
     *{2 SPACE}und{2 SPACE}* ";sw2$;" *"
                                             [4DA0]
1430 PRINT:PRINT,"betragen insgesamt = "
     ;USING"#####.##";su;:PRINT" ";si$       [4490]
1440 f$="Weiter?{2 SPACE}--> Taste dr}ck
     en!":GOSUB 320:su=0:CALL &BB18:LOCA
     TE 1,12:PRINT CHR$(20):IF p1<3 THEN
      1320                                   [1DE6]
1450 st1$="":GOTO 1250                       [4204]
1460 '                                       [9324]
1470 '***  St}ckzahlen ermitteln  ***        [DD2E]
1480 '                                       [E128]
1490 CLS:ZONE 23                             [3696]
1500 PRINT:PRINT,"Ermittlung von St}ckza
     hlen":PRINT,STRING$(26,"-")             [0E80]
1510 PRINT:PRINT,"Es kann die St}ckzahl
     zu jedem"                               [7B24]
1520 PRINT:PRINT,"beliebigen Suchwort er
     mittelt werden."                        [1DFC]
1530 PRINT:PRINT,">1<{2 SPACE}nach 1 Suc
     hwort":PRINT:PRINT,">2<{2 SPACE}nac
     h 2 Suchw!rtern"                        [A452]
1540 f$=e$:GOSUB 390:p1=p:IF p1<1 THEN |
     RETURN ELSE IF p1>2 THEN 1540           [14B8]
1550 f$="Bitte Suchwort eingeben! ":d=1:
     GOSUB 390:sw1=LEN(d$):sw1$=d$:IF sw
     1$="" THEN 1540                         [35EE]
1560 IF p1=2 THEN f$="Bitte 2.Suchwort e
     ingeben! ":d=1:GOSUB 390:sw2=LEN(d$
```

```
     ):sw2$=d$:IF sw2$="" THEN 1560          [FA16]
1570 f$="Gibt es zugeh|rige St}ckzahlen?
     "+h$:GOSUB 320:GOSUB 480:IF t$="J"
     THEN k1$="*" ELSE k1$="":GOTO 1590      [633A]
1580 f$="Die zugeh|rigen St}ckzahlen ste
     hen in welcher Spalte?{2 SPACE}Nr."
     :GOSUB 390:sz=p:IF sz<1 THEN 1570 E
     LSE IF sz>s THEN 1580                   [0EDA]
1590 IF p1=1 THEN f$="Suchwort{2 SPACE}*
      "+sw1$+" *{2 SPACE}-{2 SPACE}St}ck
     zahl wird ermittelt!"+i$:GOTO 1610
     ELSE g$="Suchw|rter{2 SPACE}* "+sw1
     $+" *{2 SPACE}und{2 SPACE}* "+sw2$+
     " *{2 SPACE}-{2 SPACE}"                 [FFD8]
1600 IF sw1+sw2 <19 THEN f$=g$+"St}ckzah
     len werden ermittelt!"+CHR$(10)+CHR
     $(13)+"Bitte warten ...." ELSE f$=g
     $+CHR$(10)+CHR$(13)+"St}ckzahlen we
     rden ermittelt!"+i$                     [68CC]
1610 GOSUB 320:FOR z=1 TO z1:GOSUB 170:s
     w=sw1:sw$=sw1$:GOSUB 540                [DEF2]
1620 IF swa=1 THEN IF k1$="*" THEN GOSUB
      790 ELSE su=su+1                       [E3B6]
1630 NEXT:PRINT CHR$(7):LOCATE 1,14          [DCDA]
1640 IF p1=1 THEN PRINT,"Die St}ckzahl z
     um Suchwort{2 SPACE}* ";sw1$;" *" E
     LSE PRINT,"Die St}ckzahl zu den Suc
     hw|rtern":PRINT:PRINT,"* "+sw1$+" *
     {2 SPACE}und{2 SPACE}* "+sw2$+" *"      [D700]
1650 PRINT:PRINT,"betr{gt insgesamt = ";
     su;"St}ck.":f$="Weiter?{2 SPACE}-->
      Taste dr}cken!":GOSUB 320:CALL &BB
     18:LOCATE 1,14:PRINT CHR$(20)           [9B0C]
1660 su=0:k1$="":GOTO 1550                   [5D36]
```

Listing 3. Inhalt der Bank 1 (Schluß)

```
10   '************************************
                                             [020C]
20   '***       MULTI - RAM DISK        ***
                                             [4810]
30   '************************************
                                             [8410]
40   '                                       [C896]
50   '************************************
                                             [4E14]
60   '***          RAMDISK.RAM          ***
                                             [5844]
70   '***             Bank 0            ***
                                             [E1EC]
80   '************************************
                                             [F11A]
90   '                                       [CEA0]
100  '***  Initialisierung  ***              [4408]
110  '                                       [DDB2]
120  IF c=1 THEN 2200                        [96CC]
130  OPENOUT"RD":MEMORY HIMEM-1:CLOSEOUT     [64C2]
140  DIM dl(10,2):e$="Programmziffer oder
      >ENTER< dr}cken! ":h$="{2 SPACE}-->
      Taste >J< dr}cken!":i$=" Bitte wart
     en ....":c$="s,z":|COMMON,@c$:|FAST:
     c=1:GOTO 2200                           [6B54]
150  '                                       [E1BA]
160  '***  RAMWRITE Daten schreiben  ***     [A548]
170  '                                       [03BE]
180  ds=1:ON s-4 GOTO 190,200,210,220,230
     ,240,250,260,270,280,290                [59EA]
190  |RAMWRITE,z,@d1$,@d2$,@d3$,@d4$,@d5$
     :RETURN                                 [6848]
200  |RAMWRITE,z,@d1$,@d2$,@d3$,@d4$,@d5$
     ,@d6$:RETURN                            [548C]
210  |RAMWRITE,z,@d1$,@d2$,@d3$,@d4$,@d5$
     ,@d6$,@d7$:RETURN                       [71E4]
220  |RAMWRITE,z,@d1$,@d2$,@d3$,@d4$,@d5$
     ,@d6$,@d7$,@d8$:RETURN                  [293E]
230  |RAMWRITE,z,@d1$,@d2$,@d3$,@d4$,@d5$
     ,@d6$,@d7$,@d8$,@d9$:RETURN             [C19A]
240  |RAMWRITE,z,@d1$,@d2$,@d3$,@d4$,@d5$
     ,@d6$,@d7$,@d8$,@d9$,@d10$:RETURN       [3D46]
250  |RAMWRITE,z,@d1$,@d2$,@d3$,@d4$,@d5$
     ,@d6$,@d7$,@d8$,@d9$,@d10$,@d11$:RET
     URN                                     [3EF4]
260  |RAMWRITE,z,@d1$,@d2$,@d3$,@d4$,@d5$
     ,@d6$,@d7$,@d8$,@d9$,@d10$,@d11$,@d1
     2$:RETURN                               [0AA4]
270  |RAMWRITE,z,@d1$,@d2$,@d3$,@d4$,@d5$
     ,@d6$,@d7$,@d8$,@d9$,@d10$,@d11$,@d1
     2$,@d13$:RETURN                         [7456]
280  |RAMWRITE,z,@d1$,@d2$,@d3$,@d4$,@d5$
     ,@d6$,@d7$,@d8$,@d9$,@d10$,@d11$,@d1
     2$,@d13$,@d14$:RETURN                   [200A]
290  |RAMWRITE,z,@d1$,@d2$,@d3$,@d4$,@d5$
     ,@d6$,@d7$,@d8$,@d9$,@d10$,@d11$,@d1
     2$,@d13$,@d14$,@d15$:RETURN             [C9C0]
300  '                                       [E2B4]
310  '***  RAMREAD  Daten lesen  ***         [D56C]
320  '                                       [E0B8]
330  ON s-4 GOTO 340,350,360,370,380,390,
     400,410,420,430,440                     [11FE]
340  |RAMREAD,z,@d1$,@d2$,@d3$,@d4$,@d5$:
     RETURN                                  [BE64]
350  |RAMREAD,z,@d1$,@d2$,@d3$,@d4$,@d5$,
     @d6$:RETURN                             [97BA]
360  |RAMREAD,z,@d1$,@d2$,@d3$,@d4$,@d5$,
     @d6$,@d7$:RETURN                        [4812]
370  |RAMREAD,z,@d1$,@d2$,@d3$,@d4$,@d5$,
     @d6$,@d7$,@d8$:RETURN                   [AE6C]
380  |RAMREAD,z,@d1$,@d2$,@d3$,@d4$,@d5$,
     @d6$,@d7$,@d8$,@d9$:RETURN              [6DC8]
390  |RAMREAD,z,@d1$,@d2$,@d3$,@d4$,@d5$,
     @d6$,@d7$,@d8$,@d9$,@d10$:RETURN        [7A74]
400  |RAMREAD,z,@d1$,@d2$,@d3$,@d4$,@d5$,
     @d6$,@d7$,@d8$,@d9$,@d10$,@d11$:RETU
     RN                                      [9210]
410  |RAMREAD,z,@d1$,@d2$,@d3$,@d4$,@d5$,
     @d6$,@d7$,@d8$,@d9$,@d10$,@d11$,@d12
     $:RETURN                                [42C0]
420  |RAMREAD,z,@d1$,@d2$,@d3$,@d4$,@d5$,
     @d6$,@d7$,@d8$,@d9$,@d10$,@d11$,@d12
     $,@d13$:RETURN                          [B572]
430  |RAMREAD,z,@d1$,@d2$,@d3$,@d4$,@d5$,
     @d6$,@d7$,@d8$,@d9$,@d10$,@d11$,@d12
     $,@d13$,@d14$:RETURN                    [F626]
440  |RAMREAD,z,@d1$,@d2$,@d3$,@d4$,@d5$,
     @d6$,@d7$,@d8$,@d9$,@d10$,@d11$,@d12
     $,@d13$,@d14$,@d15$:RETURN              [40DC]
450  RETURN                                  [AB32]
460  '                                       [E2C2]
470  '***  WINDOW 1  ***                     [470E]
480  '                                       [70C6]
490  WINDOW 1,80,1,3:PEN 0:PAPER 1:CLS       [F384]
500  LOCATE 22,2:PRINT"* *{2 SPACE}M U L
     T I{2 SPACE}-{2 SPACE}R A M{3 SPACE}
     D I S K{2 SPACE}* *"                    [4F0A]
510  PRINT" Datei:{2 SPACE}";USING"\{18 S
     PACE}\";n$;:PRINT" vom ";USING"\{8 S
     PACE}\";da$;:PRINT"{6 SPACE}***{6 SP
     ACE}Dateikarten:";dk                    [8DFE]
520  WINDOW 1,80,4,22:CALL &BB9C             [BF00]
530  RETURN                                  [CC30]
540  '                                       [E2C0]
550  '***  WINDOW 2  ***                     [420E]
560  '                                       [E4C4]
570  WINDOW 1,80,23,25:CALL &BB9C:CLS        [BBAA]
580  PRINT:PRINT f$                          [E076]
590  WINDOW 1,80,4,22:CALL &BB9C             [240E]
600  RETURN                                  [9B2C]
610  '                                       [0DBC]
620  '***  WINDOW 2 - INPUT  ***             [C104]
630  '                                       [0FC0]
640  WINDOW 1,80,23,25:CALL &BB9C:CLS        [8AA6]
650  PRINT:PRINT f$+" ";                     [8206]
660  IF d=1 THEN LINE INPUT d$:IF LEN(d$)
      >20 THEN 640 ELSE 680                  [A03A]
670  INPUT"",p                               [BB5A]
680  WINDOW 1,80,4,22:CALL &BB9C:d=0         [4224]
690  RETURN                                  [B13E]
700  '                                       [0ABC]
710  '***  INKEY - ABFRAGE  ***              [0160]
720  '                                       [08C0]
730  t$=INKEY$:IF t$="" THEN 730             [037C]
740  t$=UPPER$(t$)                           [9052]
750  RETURN                                  [BE38]
760  '                                       [E4C8]
770  '***  Daten den einzelnen RAM -  ***
                                             
780  '***  Stringvariablen }bergeben  ***    [16B2]
                                             [EEC8]
790  '                                       [71CE]
800  ON s1 GOTO 810,820,830,840,850,860,8
     70,880,890,900,910,920,930,940,950      [0D48]
810  d1$=d$:bb=b1:RETURN                     [873E]
820  d2$=d$:bb=b2:RETURN                     [5844]
830  d3$=d$:bb=b3:RETURN                     [814A]
840  d4$=d$:bb=b4:RETURN                     [9A50]
850  d5$=d$:bb=b5:RETURN                     [3356]
860  d6$=d$:bb=b6:RETURN                     [945C]
870  d7$=d$:bb=b7:RETURN                     [7D62]
880  d8$=d$:bb=b8:RETURN                     [6668]
890  d9$=d$:bb=b9:RETURN                     [7F6E]
900  d10$=d$:bb=b10:RETURN                   [64FE]
910  d11$=d$:bb=b11:RETURN                   [B104]
920  d12$=d$:bb=b12:RETURN                   [6D0A]
930  d13$=d$:bb=b13:RETURN                   [2510]
940  d14$=d$:bb=b14:RETURN                   [A516]
950  d15$=d$:bb=b15:RETURN                   [891C]
960  '                                       [B4CC]
970  '***  Zeichen pro Spalte  ***           [7446]
980  '                                       [E6D0]
990  ON s-4 GOTO 1000,1010,1020,1030,1040
     ,1050,1060,1070,1080,1090,1100          [1CFA]
1000 INPUT#x,b1,b2,b3,b4,b5:GOTO 1110        [5E4E]
1010 INPUT#x,b1,b2,b3,b4,b5,b6:GOTO 1110
```

Listing 4. »Futter« für Bank 0

```
                                                  [B7D8]
1020 INPUT#x,b1,b2,b3,b4,b5,b6,b7:GOTO 1
     110                                          [A064]
1030 INPUT#x,b1,b2,b3,b4,b5,b6,b7,b8:GOT
     O 1110                                       [D8F2]
1040 INPUT#x,b1,b2,b3,b4,b5,b6,b7,b8,b9:
     GOTO 1110                                    [1A82]
1050 INPUT#x,b1,b2,b3,b4,b5,b6,b7,b8,b9,
     b10:GOTO 1110                                [DD62]
1060 INPUT#x,b1,b2,b3,b4,b5,b6,b7,b8,b9,
     b10,b11:GOTO 1110                            [DE44]
1070 INPUT#x,b1,b2,b3,b4,b5,b6,b7,b8,b9,
     b10,b11,b12:GOTO 1110                        [2F28]
1080 INPUT#x,b1,b2,b3,b4,b5,b6,b7,b8,b9,
     b10,b11,b12,b13:GOTO 1110                    [F20E]
1090 INPUT#x,b1,b2,b3,b4,b5,b6,b7,b8,b9,
     b10,b11,b12,b13,b14:GOTO 1110                [3CF6]
1100 INPUT#x,b1,b2,b3,b4,b5,b6,b7,b8,b9,
     b10,b11,b12,b13,b14,b15:GOTO 1110            [F2CE]
1110 r=b1+b2+b3+b4+b5                             [864E]
1120 IF s>5 THEN r=r+b6                           [8798]
1130 IF s>6 THEN r=r+b7                           [3F9E]
1140 IF s>7 THEN r=r+b8                           [AFA4]
1150 IF s>8 THEN r=r+b9                           [5FAA]
1160 IF s>9 THEN r=r+b10                          [D3FE]
1170 IF s>10 THEN r=r+b11                         [9952]
1180 IF s>11 THEN r=r+b12                         [5E58]
1190 IF s>12 THEN r=r+b13                         [635E]
1200 IF s>13 THEN r=r+b14                         [4752]
1210 IF s>14 THEN r=r+b15                         [7658]
1220 '                                            [B718]
1230 '***  RAM DISK oeffnen und struktur
     ieren  ***                                   [D4C6]
1240 '                                            [911C]
1250 |RAMOPEN,r:ON s-4 GOTO 1260,1270,12
     80,1290,1300,1310,1320,1330,1340,13
     50,1360                                      [6F56]
1260 |RAMFIELD,b1,b2,b3,b4,b5:GOTO 1370          [A798]
1270 |RAMFIELD,b1,b2,b3,b4,b5,b6:GOTO 13
     70                                           [5222]
1280 |RAMFIELD,b1,b2,b3,b4,b5,b6,b7:GOTO
      1370                                        [6BAE]
1290 |RAMFIELD,b1,b2,b3,b4,b5,b6,b7,b8:G
     OTO 1370                                     [CB3C]
1300 |RAMFIELD,b1,b2,b3,b4,b5,b6,b7,b8,b
     9:GOTO 1370                                  [2CBA]
1310 |RAMFIELD,b1,b2,b3,b4,b5,b6,b7,b8,b
     9,b10:GOTO 1370                              [6F9A]
1320 |RAMFIELD,b1,b2,b3,b4,b5,b6,b7,b8,b
     9,b10,b11:GOTO 1370                          [207C]
1330 |RAMFIELD,b1,b2,b3,b4,b5,b6,b7,b8,b
     9,b10,b11,b12:GOTO 1370                      [7D60]
1340 |RAMFIELD,b1,b2,b3,b4,b5,b6,b7,b8,b
     9,b10,b11,b12,b13:GOTO 1370                  [3446]
1350 |RAMFIELD,b1,b2,b3,b4,b5,b6,b7,b8,b
     9,b10,b11,b12,b13,b14:GOTO 1370              [E72E]
1360 |RAMFIELD,b1,b2,b3,b4,b5,b6,b7,b8,b
     9,b10,b11,b12,b13,b14,b15:GOTO 1370
                                                  [F018]
1370 |RECORDS:dk=rec-1:rec=0                      [0270]
1380 RETURN                                       [B898]
1390 '                                            [BA28]
1400 '***  Bildschirmanzeige  ***                 [5CD8]
1410 '                                            [BA1A]
1420 CLS:ZONE 22:PRINT CHR$(7)                    [EC66]
1430 PRINT">";z;"/{2 SPACE}1 <",b$(1),d1
     $                                            [3180]
1440 PRINT">";z;"/{2 SPACE}2 <",b$(2),d2
     $                                            [7988]
1450 PRINT">";z;"/{2 SPACE}3 <",b$(3),d3
     $                                            [9190]
1460 PRINT">";z;"/{2 SPACE}4 <",b$(4),d4
     $                                            [0198]
1470 PRINT">";z;"/{2 SPACE}5 <",b$(5),d5
     $                                            [F1A0]
1480 IF s>5 THEN PRINT">";z;"/{2 SPACE}6
      <",b$(6),d6$                                [50B0]
1490 IF s>6 THEN PRINT">";z;"/{2 SPACE}7
      <",b$(7),d7$                                [99BA]
1500 IF s>7 THEN PRINT">";z;"/{2 SPACE}8
      <",b$(8),d8$                                [C8B2]
1510 IF s>8 THEN PRINT">";z;"/{2 SPACE}9
      <",b$(9),d9$                                [8BBC]
1520 IF s>9 THEN PRINT">";z;"/ 10 <",b$(
     10),d10$                                     [B470]
1530 IF s>10 THEN PRINT">";z;"/ 11 <",b$
     (11),d11$                                    [FBC8]
1540 IF s>11 THEN PRINT">";z;"/ 12 <",b$
     (12),d12$                                    [26D2]
1550 IF s>12 THEN PRINT">";z;"/ 13 <",b$
     (13),d13$                                    [E5DC]
1560 IF s>13 THEN PRINT">";z;"/ 14 <",b$
     (14),d14$                                    [BCE6]
1570 IF s>14 THEN PRINT">";z;"/ 15 <",b$
     (15),d15$                                    [57F0]
1580 RETURN                                       [BE9C]
1590 '                                            [BE2C]
1600 '***  aus RAM DISK ausgelesene Date
     n sichern  ***                               [33CE]
1610 '                                            [BE1E]
1620 d1$=d1$:d2$=d2$:d3$=d3$:d4$=d4$:d5$
     =d5$                                         [61A0]
1630 IF s>5 THEN d6$=d6$                          [A84E]
1640 IF s>6 THEN d7$=d7$                          [CA56]
1650 IF s>7 THEN d8$=d8$                          [7C5E]
1660 IF s>8 THEN d9$=d9$                          [A666]
1670 IF s>9 THEN d10$=d10$                        [A80A]
1680 IF s>10 THEN d11$=d11$                       [4860]
1690 IF s>11 THEN d12$=d12$                       [5068]
1700 IF s>12 THEN d13$=d13$                       [1D5E]
1710 IF s>13 THEN d14$=d14$                       [4166]
1720 IF s>14 THEN d15$=d15$                       [716E]
1730 RETURN                                       [8A96]
1740 '                                            [9726]
1750 '***  nach Suchwort suchen  ***              [E69E]
1760 '                                            [952A]
1770 IF sw$=LEFT$(d1$,sw) THEN 1930               [C028]
1780 IF sw$=LEFT$(d2$,sw) THEN 1930               [FF2C]
1790 IF sw$=LEFT$(d3$,sw) THEN 1930               [B230]
1800 IF sw$=LEFT$(d4$,sw) THEN 1930               [5C22]
1810 IF sw$=LEFT$(d5$,sw) THEN 1930               [F526]
1820 IF sw$=LEFT$(d6$,sw) THEN 1930               [E22A]
1830 IF sw$=LEFT$(d7$,sw) THEN 1930               [C72E]
1840 IF sw$=LEFT$(d8$,sw) THEN 1930               [7032]
1850 IF sw$=LEFT$(d9$,sw) THEN 1930               [C936]
1860 IF sw$=LEFT$(d10$,sw) THEN 1930              [1988]
1870 IF sw$=LEFT$(d11$,sw) THEN 1930              [7D8C]
1880 IF sw$=LEFT$(d12$,sw) THEN 1930              [BD90]
1890 IF sw$=LEFT$(d13$,sw) THEN 1930              [9194]
1900 IF sw$=LEFT$(d14$,sw) THEN 1930              [7B86]
1910 IF sw$=LEFT$(d15$,sw) THEN 1930              [E78A]
1920 swa=0:RETURN                                 [EB7C]
1930 swa=1:RETURN                                 [5280]
1940 '                                            [E32A]
1950 '***  Dateinamen ausdrucken  ***             [EABA]
1960 '                                            [252E]
1970 f$="Dateiname wird ausgedruckt!"+i$
     :GOSUB 550                                   [A0D2]
1980 PRINT#8,CHR$(18);CHR$(14);TAB(22-IN
     T(LEN(n$)/2)) n$:PRINT#8:PRINT#8:PR
     INT#8:zz=zz+4                                [A6B8]
1990 RETURN                                       [8CA6]
2000 '                                            [BA12]
2010 '***  Spaltenbezeichnungen ausdruck
     en  ***                                      [3C2E]
2020 '                                            [BC16]
2030 f$="Spaltenbezeichnungen werden aus
     gedruckt!"+i$:GOSUB 550                      [47C0]
2040 ZONE 22:PRINT#8,CHR$(15);:IF s<11 T
     HEN s2=s ELSE s2=10                          [5FA2]
2050 PRINT#8,TAB(12)"lfd.Nr.{4 SPACE}";:
     FOR s1=1 TO 5:PRINT#8,b$(s1),:NEXT:
     zz=zz+1                                      [E2D6]
2060 IF s>5 THEN PRINT#8,TAB(22)" ";:FOR
      s1=6 TO s2:PRINT#8,b$(s1),:NEXT:zz
     =zz+1                                        [2F68]
2070 IF s>10 THEN PRINT#8,TAB(22)" ";:FO
     R s1=11 TO s:PRINT#8,b$(s1),:NEXT:z
     z=zz+1                                       [9FB6]
2080 PRINT#8,TAB(12) STRING$(121,"-"):zz
     =zz+1                                        [45C2]
2090 RETURN                                       [FB96]
2100 '                                            [BC14]
2110 '***  Dateikarten ausdrucken  ***            [0290]
2120 '                                            [9A18]
2130 ZONE 22:PRINT#8,CHR$(15);                    [6BCE]
2140 PRINT#8,TAB(12)"Nr.";z;TAB(22)" ";d
     1$,d2$,d3$,d4$,d5$:zz=zz+1                   [4FCA]
2150 IF s>5 THEN PRINT#8,TAB(22)" ";d6$,
     d7$,d8$,d9$,d10$:zz=zz+1                     [B6FA]
2160 IF s>10 THEN PRINT#8,TAB(22)" ";d11
     $,d12$,d13$,d14$,d15$:zz=zz+1                [00BE]
2170 IF dz=1 THEN PRINT#8:zz=zz+1                 [FEEE]
2180 RETURN                                       [C096]
2190 '                                            [2526]
2200 '***  Hauptmen}  ***                         [3B8C]
2210 '                                            [BD18]
2220 s=ss:p1=0:ZONE 23:GOSUB 470:f$="":G
     OSUB 550                                     [F3DA]
2230 CLS:PRINT:PRINT STRING$(80,"-"):PRI
     NT,"* * *{3 SPACE}H A U P T M E N }
     {2 SPACE}* * *":PRINT:PRINT STRING$
     (80,"-")                                     [0AB2]
2240 PRINT,"> 1 <{2 SPACE}Datei einlesen
     "                                            [EDD4]
2250 PRINT,"> 2 <{2 SPACE}Datei anlegen"
                                                  [4AE6]
2260 PRINT,"> 3 <{2 SPACE}Dateispalten b
     ezeichnen"                                   [EE5A]
2270 PRINT,"> 4 <{2 SPACE}Dateikarten be
     schreiben"                                   [EB38]
2280 PRINT,"> 5 <{2 SPACE}Dateikarten {n
     dern/l|schen"                                [8BBC]
2290 PRINT,"> 6 <{2 SPACE}Dateikarten an
     sehen,suchen usw."                           [0ACE]
2300 PRINT,"> 7 <{2 SPACE}Dateikarten au
     sdrucken"                                    [09A8]
2310 PRINT,"> 8 <{2 SPACE}Datei sichern"
                                                  [8910]
2320 PRINT,"> 9 <{2 SPACE}RAM DISK schli
     e~en"                                        [6C56]
```

```
2330 LOCATE 26,19:PRINT"(c) 1986 by kaba
     soft-computing"                        [A4BC]
2340 f$=e$:GOSUB 620:IF p<1 OR p>9 THEN
     2340                                   [54E2]
2350 IF p>2 AND rd=0 THEN f$="Neue Datei
      anlegen?"+h$:GOSUB 550:GOSUB 710:I
     F t$="J" THEN 2610 ELSE 2340           [3BD2]
2360 ON p GOTO 2380,2610,2800,2900,3070,
     3550,3970,4340,4810                    [143C]
2370 '                                      [2726]
2380 '***  Datei einlesen  ***              [CFD4]
2390 '                                      [992A]
2400 IF rd=1 THEN f$="RAM DISK belegt! N
     eue Datei wirklich einlesen?"+h$:GO
     SUB 550:GOSUB 710:IF t$="J" THEN ER
     ASE b$ ELSE 2340                       [B16A]
2410 n$="":da$="":rd=0:dk=0:b=0:GOSUB 47
     0                                      [7B1E]
2420 CLS:f$="Datei wird eingelesen!"+i$:
     GOSUB 550                              [C5B0]
2430 PRINT:OPENIN "                         [D976]
2440 LINE INPUT#9,n$:LINE INPUT#9,da$:IN
     PUT#9,s,z1,lz:DIM b$(s):b=1            [DC2A]
2450 FOR s1=1 TO s:LINE INPUT#9,b$(s1):N
     EXT                                    [F9D0]
2460 FOR z2=0 TO 10:INPUT#9,dl(z2,1),dl(
     z2,2):NEXT                             [0238]
2470 x=9:GOSUB 970:x=0:GOSUB 470:LOCATE
     1,3                                    [4A16]
2480 FOR z=1 TO z1:LINE INPUT#9,d1$:LINE
      INPUT#9,d2$:LINE INPUT#9,d3$:LINE
     INPUT#9,d4$:LINE INPUT#9,d5$           [9528]
2490 IF s>5 THEN LINE INPUT#9,d6$           [2A22]
2500 IF s>6 THEN LINE INPUT#9,d7$           [CA16]
2510 IF s>7 THEN LINE INPUT#9,d8$           [831C]
2520 IF s>8 THEN LINE INPUT#9,d9$           [A822]
2530 IF s>9 THEN LINE INPUT#9,d10$          [7876]
2540 IF s>10 THEN LINE INPUT#9,d11$         [D8CA]
2550 IF s>11 THEN LINE INPUT#9,d12$         [86D0]
2560 IF s>12 THEN LINE INPUT#9,d13$         [DCD6]
2570 IF s>13 THEN LINE INPUT#9,d14$         [D2DC]
2580 IF s>14 THEN LINE INPUT#9,d15$         [E0E2]
2590 GOSUB 160:NEXT:CLOSEIN:rd=1:ds=0:ss
     =s:|GOSUB,1,100:GOTO 2200              [CBB4]
2600 '                                      [961E]
2610 '***  Datei anlegen  ***               [BBDA]
2620 '                                      [9422]
2630 IF rd=1 THEN f$="RAM DISK belegt! N
     eue Datei wirklich anlegen?"+h$:GOS
     UB 550:GOSUB 710:IF t$="J" THEN ERA
     SE b$ ELSE 2340                        [1E82]
2640 n$="":da$="":rd=0:dk=0:b=0:z1=0:GOS
     UB 470:CLS:ZONE 17                     [B400]
2650 f$="Neue Datei anlegen.{2 SPACE}-{2
      SPACE}Bitte Daten eingeben!":GOSUB
      550                                   [3FAE]
2660 PRINT:PRINT,"Untermen}{2 SPACE}> NE
     UE DATEI ANLEGEN <":PRINT,STRING$(3
     3,"-")                                 [860E]
2670 PRINT:PRINT,"Zum Anlegen der neuen
     Datei werden":PRINT,"folgende Angab
     en ben|tigt!"                          [70F8]
2680 PRINT:PRINT,"Datum ........{4 SPACE
     }";:LINE INPUT da$                     [E02A]
2690 PRINT:PRINT,"Dateiname ....{4 SPACE
     }";:LINE INPUT n$:IF LEN(n$) >20 TH
     EN PRINT CHR$(11);CHR$(11);CHR$(20)
     ;:GOTO 2690                            [DB98]
2700 PRINT:PRINT,"Wieviel Spalten (5 bis
      15) soll die Datei haben";:INPUT s
                                            [98D8]
2710 IF s<5 OR s>15 THEN PRINT CHR$(11);
     CHR$(11);CHR$(20);:GOTO 2700           [6674]
2720 PRINT:PRINT,"Wieviel Zeichen (1 bis
      20) soll jede Spalte haben?"          [8CC8]
2730 PRINT,"(z.B. 20,16,10,10,7,7,6,20,4
     ,20 )":PRINT,"--> ";:GOSUB 970         [92D2]
2740 f$="RAM DISK wird ge|ffnet!{2 SPACE
     }Bitte warten ....":GOSUB 550:n$=UP
     PER$(n$):ss=s:DIM b$(s)                [EB68]
2750 CLS:LOCATE 22,4:PRINT"* *{2 SPACE}M
      U L T I{2 SPACE}-{2 SPACE}R A M{3
     SPACE}D I S K{2 SPACE}* *"             [B1B8]
2760 PRINT:PRINT:PRINT,"{2 SPACE}Datei:{
     2 SPACE}";USING"\{18 SPACE}\";n$;:P
     RINT" vom ";da$                        [97C2]
2770 LOCATE 16,9:PRINT"ist dimensioniert
      f}r";dk;"Dateikarten mit";s;"Spalt
     en":LOCATE 21,11:PRINT"bei durchsch
     nittlich";CINT(r/s);"Zeichen je Spa
     lte"                                   [B4FA]
```

Listing 4. »Futter« für Bank 0 (Fortsetzung)

```
2780 FOR t=1 TO 3000:NEXT:rd=1:|GOSUB,1,
     100:GOTO 2200                         [3E3C]
2790 '                                     [E132]
2800 '***  Bezeichnungen eingeben  ***     [585E]
2810 '                                     [C924]
2820 IF b=1 THEN f$="Bezeichnungen sind
     schon eingegeben!":GOSUB 550:FOR t=
     1 TO 1500:NEXT:GOTO 2340              [8E24]
2830 f$="Bezeichnung der Dateispalte ein
     geben":GOSUB 550                      [374C]
2840 CLS:d$="":PRINT:FOR s1=1 TO s         [5E06]
2850 PRINT"> "; USING"##";s1;:PRINT" <",
     :LINE INPUT d$:IF LEN(d$) >20 THEN
     PRINT CHR$(11);CHR$(20);:GOTO 2850    [605A]
2860 IF s1=1 AND d$="" THEN 2840           [AB14]
2870 IF d$<>"" THEN b$(s1)=d$              [E22A]
2880 NEXT:b=1:GOTO 2200                    [7424]
2890 '                                     [A134]
2900 '***  Dateikarten beschreiben  ***    [6328]
2910 '                                     [2D26]
2920 FOR z2=0 TO 10:IF dl(z2,2)=0 THEN N
     EXT                                   [9C00]
2930 IF z2=11 THEN a=z1+1:e=dk:GOTO 2950
                                           [9614]
2940 a=dl(z2,1):e=dl(z2,2)                 [4A7C]
2950 FOR z=a TO e:f$="Dateikarte Nr."+ST
     R$(z)+"{2 SPACE}-{2 SPACE}Bitte Dat
     en eingeben!":GOSUB 550               [7E9A]
2960 CLS:PRINT:FOR s1=1 TO s               [6E86]
2970 PRINT">";z;"/ ";USING"##";s1;:PRINT
     " <",b$(s1),:LINE INPUT d$:GOSUB 77
     0:IF d1$="" THEN 2200                 [2456]
2980 IF LEN(d$) >bb THEN PRINT CHR$(11);
     CHR$(20):LOCATE 1,18:PRINT"Dateispa
     lte ist nur f}r"+STR$(bb)+" Zeichen
      dimensioniert!{2 SPACE}-{2 SPACE}B
     itte neu eingeben!":LOCATE 1,s1+1:G
     OTO 2970                              [D068]
2990 PRINT CHR$(20);:NEXT                  [4912]
3000 GOSUB 160:IF d1$>"" THEN IF z2=11 T
     HEN z1=z1+1 ELSE dl(z2,1)=z+1         [F356]
3010 f$="Weitere Dateikarte anlegen?"+h$
     :GOSUB 550:GOSUB 710:IF t$="J" AND
     d1$>"" THEN NEXT                      [E900]
3020 IF z2=11 THEN 3040                    [772A]
3030 IF z>e THEN dl(z2,1)=0:dl(z2,2)=0     [A714]
3040 IF z>dk THEN f$="{5 SPACE}* * *{2 S
     PACE}RAM DISK ist belegt!{2 SPACE}*
      * *":GOSUB 550:FOR t=1 TO 2000:NEX
     T:GOTO 2200                           [5A1C]
3050 IF t$="J" THEN 2920 ELSE 2200         [F6C6]
3060 '                                     [9520]
3070 '***  Untermen}  > [NDERN und L\SCH
     EN <  ***                             [3388]
3080 '                                     [BF24]
3090 CLS                                   [329C]
3100 PRINT:PRINT,"Untermen}{2 SPACE}> [N
     DERN und L\SCHEN <":PRINT,STRING$(3
     3,"-")                                [3242]
3110 PRINT:PRINT,"Was soll ge{ndert oder
      gel|scht werden?"                    [D6CC]
3120 PRINT:PRINT,">1<{2 SPACE}Datum der
     Datei {ndern"                         [6648]
3130 PRINT:PRINT,">2<{2 SPACE}Spaltenbez
     eichnung {ndern"                      [8364]
3140 PRINT:PRINT,">3<{2 SPACE}Dateikarte
      {ndern"                              [F092]
3150 PRINT:PRINT,">4<{2 SPACE}Dateikarte
      l|schen"                             [A464]
3160 f$=e$:GOSUB 620:IF p<1 THEN 2200 EL
     SE IF p>4 THEN 3160                   [C7B0]
3170 ON p GOTO 3190,3250,3320,3440         [E27E]
3180 '                                     [BD26]
3190 '***  Datum der Datei {ndern  ***     [A33E]
3200 '                                     [9318]
3210 CLS:f$="Datum der Datei wirklich {n
     dern?"+h$:GOSUB 550:GOSUB 710         [4BEA]
3220 IF t$="J" THEN f$="Aktuelles Datum
     eingeben!":d=1:GOSUB 620:da$=d$:GOS
     UB 470:ds=1                           [0920]
3230 GOTO 3090                             [781A]
3240 '                                     [B720]
3250 '***  Bezeichnung {ndern  ***         [8B62]
3260 '                                     [9524]
3270 CLS:PRINT:FOR s1=1 TO s:PRINT"> ";U
     SING"##"; s1;:PRINT" <",b$(s1):NEXT
                                           [B866]
3280 f$="Welche Spaltenbezeichnung {nder
     n?{2 SPACE}Nr.":GOSUB 620:s1=p:IF s
     1<1 THEN 3070 ELSE IF s1>s THEN 328
     0                                     [B1A2]
3290 f$="Neue Bezeichnung der Spalte"+ST
     R$(s1)+":":d=1:GOSUB 620:b$(s1)=d$:
     ds=1:PRINT CHR$(7)                    [5792]
3300 GOTO 3270                             [8516]
3310 '                                     [901C]
3320 '***  Dateikarten {ndern  ***         [F552]
3330 '                                     [9220]
3340 CLS:f$="Welche Dateikarte {ndern?{2
      SPACE}Nr.":GOSUB 620:z=p:IF z<1 TH
     EN 3070 ELSE IF z>z1 THEN 3340        [556C]
3350 GOSUB 310:GOSUB 1600:GOSUB 1400       [29BE]
3360 f$="Dateikarte Nr."+STR$(z)+" wirkl
     ich {ndern?"+h$:GOSUB 550:GOSUB 710
     :IF t$<>"J" THEN 3410                 [7FA0]
3370 f$="Welche Spalte {ndern?{2 SPACE}N
     r.":GOSUB 620:s1=p:IF s1<1 THEN 341
     0 ELSE IF s1>s THEN 3370              [069E]
3380 f$="Neue Dateneingabe Spalte"+STR$(
     s1)+":":d=1:GOSUB 620:GOSUB 770       [D084]
3390 IF LEN(d$) >bb THEN LOCATE 1,18:PRI
     NT"Dateispalte ist nur f}r"+STR$(bb
     )+" Zeichen dimensioniert!{2 SPACE}
     -{2 SPACE}Bitte neu eingeben!":GOTO
      3380                                 [5666]
3400 GOSUB 160:GOSUB 1400:GOTO 3370        [5E3A]
3410 f$="N{chste Dateikarte Nr."+STR$(z+
     1)+" {ndern?"+h$:GOSUB 550:GOSUB 71
     0:IF t$="J" THEN z=z+1:IF z<=z1 THE
     N 3350                                [E7E8]
3420 ds=1:GOTO 3070                        [0616]
3430 '                                     [C222]
3440 '***  Dateikarten loeschen  ***       [B0D6]
3450 '                                     [E426]
3460 CLS:g$="Welche Dateikarte(n) l|sche
     n?{2 SPACE}Von Nr.":f$=g$:GOSUB 620
     :a=p:IF a<1 THEN 3070 ELSE IF a>z1
     THEN 3460                             [75EE]
3470 f$=g$+STR$(a)+" bis Nr.":GOSUB 620:
     e=p:IF e<a THEN e=a ELSE IF e>z1 TH
     EN e=z1                               [81A4]
3480 g$="Dateikarten Nr."+STR$(a)+" bis
     Nr."+STR$(e):f$=g$+" wirklich l|sch
     en?"+h$:GOSUB 550:GOSUB 710:IF t$<>
     "J" THEN 3070                         [F0EE]
3490 f$=g$+" werden gel|scht!"+i$:GOSUB
     550:g$=""                             [DCDE]
3500 FOR z=a TO e:GOSUB 310:d1$="":d2$="
     ":d3$="":d4$="":d5$="":d6$="":d7$="
     ":d8$="":d9$="":d10$="":d11$="":d12
     $="":d13$="":d14$="":d15$="":GOSUB
     160:NEXT                              [DD8C]
3510 IF e=z1 THEN z1=a-1:GOTO 3070         [19B0]
3520 dl(lz,1)=a:dl(lz,2)=e:lz=lz+1:IF lz
     >10 THEN lz=0                         [B4F8]
3530 GOTO 3070                             [5A1C]
3540 '                                     [9926]
3550 '***  Untermen}  > ANSEHEN, SUCHEN
     und EXTRAS < ***                      [C84E]
3560 '                                     [972A]
3570 CLS:ZONE 20                           [AE92]
3580 PRINT:PRINT,"Untermen}{2 SPACE}> AN
     SEHEN, SUCHEN und EXTRAS <":PRINT,S
     TRING$(41,"-")                        [7358]
3590 PRINT:PRINT,">1< Dateikarten nach N
     r. ansehen"                           [9E6C]
3600 PRINT:PRINT,">2< Dateikarten nach S
     uchwort ansehen                       [E3FC]
3610 PRINT:PRINT,">3< Inventarsummen der
      Datei ermitteln"                     [AA5A]
3620 PRINT:PRINT,">4< St}ckzahlen der Da
     tei ermitteln"                        [17CE]
3630 f$=e$:GOSUB 620:IF p<1 THEN 2200 EL
     SE IF p>4 THEN 3630                   [CFB8]
3640 ON p GOTO 3660,3740,3890,3930         [67AE]
3650 '                                     [982A]
3660 '***  Datei-Karten ansehen  ***       [4A1A]
3670 '                                     [962E]
3680 CLS                                   [8FA6]
3690 g$="Welche Dateikarte(n) ansehen?{2
      SPACE}Von Nr.":f$=g$:GOSUB 620:a=p
     :IF a<1 THEN 3550 ELSE IF a>z1 THEN
      3690                                 [15A2]
3700 f$=g$+STR$(a)+" bis Nr.":GOSUB 620:
     e=p:IF e<a THEN e=a ELSE IF e>z1 TH
     EN e=z1                               [A09C]
3710 FOR z=a TO e:GOSUB 310:GOSUB 1400:f
     $="Dateikarte Nr."+STR$(z)+"{2 SPAC
     E}-{2 SPACE}Weiter?{2 SPACE}--> Tas
     te dr}cken!":GOSUB 550:CALL &BB18:N
     EXT                                   [2364]
3720 g$="":GOTO 3690                       [34BA]
3730 '                                     [BA28]
3740 '***  Dateikarten suchen (und ausdr
     ucken)  ***                           [3260]
3750 '                                     [942C]
3760 CLS:f$="Bitte Suchwort eingeben! ":
     d=1:GOSUB 620:sw=LEN(d$):sw$=d$:IF
     sw$="" AND p1=3 THEN 3970 ELSE IF s
     w$="" THEN 3550                       [1A22]
3770 g$="Suchwort{2 SPACE}* "+sw$+" *{2
     SPACE}-{2 SPACE}Dateikarte wird ges
     ucht!"+i$:f$=g$:GOSUB 550             [008A]
3780 FOR z=1 TO z1:GOSUB 310:GOSUB 1750:
     IF swa=0 THEN 3840                    [56C6]
3790 IF p1=3 AND zz>59 THEN GOSUB 3870     [B48E]
3800 IF p1=3 AND zz=0 THEN GOSUB 1950:GO
     SUB 2010                              [CD34]
3810 GOSUB 1400:IF p1=3 THEN f$="Suchwor
     t{2 SPACE}* "+sw$+" *{2 SPACE}-{2 S
     PACE}Dateikarte wird ausgedruckt!":
```

```
     GOSUB 550:GOSUB 2110:GOTO 3830         [1AEE]
3820 f$="Suchwort{2 SPACE}* "+sw$+" *{2
     SPACE}-{2 SPACE}Weitersuchen?{2 SPA
     CE}--> Taste dr}cken!":GOSUB 550:CA
     LL &BB18                               [C6C0]
3830 f$=g$:GOSUB 550                        [B868]
3840 NEXT:f$="Suchwort{2 SPACE}* "+sw$+"
      *{2 SPACE}nicht vorhanden!":GOSUB
     550:IF p1=3 THEN LOCATE 1,s+3:PRINT
     "Suchwort{2 SPACE}* ";sw$;" *{2 SPA
     CE}-{2 SPACE}Datei ist vollst{ndig
     ausgedruckt!"                          [55B6]
3850 IF p1=3 AND zz>0 THEN f$="Gesamt-Au
     sdruck nach Suchwort wird beendet!"
     +i$:GOSUB 550:GOSUB 3870:g$="":GOTO
      3760                                  [CE2E]
3860 g$="":FOR t=1 TO 3000:NEXT:GOTO 376
     0                                      [6644]
3870 FOR zz=zz TO 62:PRINT#8:NEXT:PRINT#
     8,CHR$(18);TAB(26-INT(LEN(sw$)/2))"
     Dateiausdruck nach Suchwort{2 SPACE
     }* ";sw$;" *";CHR$(12):zz=0:RETURN     [99CE]
3880 '                                      [9F34]
3890 '***  Datei-Inventarsummen ermittel
     n  ***                                 [EBF6]
3900 '                                      [C526]
3910 |GOSUB,1,1180:GOTO 3550                [A638]
3920 '                                      [C32A]
3930 '***  St}ckzahlen ermitteln  ***       [E034]
3940 '                                      [292E]
3950 |GOSUB,1,1470:GOTO 3550                [5344]
3960 '                                      [E732]
3970 '***  Untermen}  > DRUCKEN <  ***      [68CE]
3980 '                                      [9D36]
3990 CLS:WIDTH 255                          [109A]
4000 PRINT:PRINT,"Untermen}{2 SPACE}> DR
     UCKEN <":PRINT,STRING$(22,"-")         [8572]
4010 PRINT:PRINT,"Was soll ausgedruckt w
     erden?"                                [6BA8]
4020 PRINT:PRINT,">1<{2 SPACE}Einzelausd
     ruck nach Nr."                         [B54C]
4030 PRINT:PRINT,">2<{2 SPACE}Gesamtausd
     ruck der Datei"                        [A978]
4040 PRINT:PRINT,">3<{2 SPACE}Gesamtausd
     ruck nach Suchwort"                    [862A]
4050 f$=e$:GOSUB 620:p1=p:IF p1<1 THEN 2
     200 ELSE IF p1>3 THEN 4050             [797E]
4060 IF dr=1 THEN 4090                      [E832]
4070 f$="Drucker angeschlossen?"+h$:GOSU
     B 550:GOSUB 710:IF t$="J" THEN PRIN
     T#8,;:dr=1                             [0A6E]
4080 IF t$<>"J" THEN f$="Bitte Drucker a
     nschlie~en!":GOSUB 550:FOR t=1 TO 1
     500:NEXT:GOTO 4070                     [0E1A]
4090 CLS:f$="Ausdruck der vollst{ndigen
     Dateikarten?"+h$:GOSUB 550:GOSUB 71
     0:IF t$="J" THEN s=ss:GOTO 4110        [3FEE]
4100 f$="Welche Dateispalten ausdrucken?
     {2 SPACE}Von Nr. 1 bis Nr.":GOSUB 6
     20:s=p:IF s<5 THEN s=5 ELSE IF s>ss
      THEN s=ss                             [0C72]
4110 IF p1>1 THEN 4140                      [42BA]
4120 f$="Ausdruck des Dateinamens?"+h$:G
     OSUB 550:GOSUB 710:IF t$="J" THEN G
     OSUB 1950                              [EE3E]
4130 f$="Ausdruck der Spaltenbezeichnung
     en?"+h$:GOSUB 550:GOSUB 710:IF t$="
     J" THEN GOSUB 2010                     [9CCC]
4140 f$="Zwischenzeile nach einer Datei-
     Karte?"+h$:GOSUB 550:GOSUB 710:IF t
     $="J" THEN dz=1 ELSE dz=0              [530E]
4150 ON p1 GOTO 4170,4270,3740              [6600]
4160 '                                      [E824]
4170 '***  Einzelausdruck  ***              [79B0]
4180 '                                      [9A28]
4190 g$="Welche Dateikarten ausdrucken?
     Von Nr.":f$=g$:GOSUB 620:a=p:IF a>z
     1 THEN 4190 ELSE IF a<1 THEN IF zz=
     0 THEN 3970 ELSE 4240                  [B5C6]
4200 f$=g$+STR$(a)+" bis Nr.":GOSUB 620:
     e=p:IF e<a THEN e=a ELSE IF e>z1 TH
     EN e=z1                                [A394]
4210 f$="Dateikarten werden ausgedruckt!
     "+i$:GOSUB 550                         [3224]
4220 FOR z=a TO e:GOSUB 310:GOSUB 1400:G
     OSUB 2110:IF zz<60 THEN NEXT           [374E]
4230 g$="":IF zz<60 THEN 4190               [0862]
4240 f$="Einzel-Ausdruck wird beendet!"+
     i$:GOSUB 550:CLS                       [EF02]
4250 FOR zz=zz TO 62:PRINT#8:NEXT:PRINT#
     8,CHR$(18);TAB(33)"Datei-Einzelausd
     ruck";CHR$(12):zz=0:GOTO 3970          [8BC2]
4260 '                                      [9A26]
4270 '***  Gesamtausdruck  ***              [36A6]
4280 '                                      [E42A]
4290 CLS:g$="Gesamtausdruck der Datei!"+
     i$:GOSUB 1950:GOSUB 2010:f$=g$:GOSU
     B 550                                  [79B6]
4300 FOR z=1 TO z1:GOSUB 310:GOSUB 1400:
     GOSUB 2110:IF z<z1 AND zz>59 THEN 4
     320                                    [1A88]
4310 NEXT:PRINT:PRINT"Datei{2 SPACE}* ";
     n$;" *{2 SPACE}ist vollst{ndig ausg
     edruckt!"                              [F4A8]
4320 FOR zz=zz TO 62:PRINT#8:NEXT:PRINT#
     8,CHR$(18);TAB(33)"Datei-Gesamtausd
     ruck";CHR$(12):zz=0:IF z<z1 THEN CL
     S:GOSUB 1950:GOSUB 2010:f$=g$:GOSUB
      550:GOTO 4310 ELSE g$="":GOTO 3970
                                            [CD14]
4330 '                                      [BD22]
4340 '***  Untermen}  >SICHERN <  ***       [937E]
4350 '                                      [9B26]
4360 CLS                                    [189E]
4370 PRINT:PRINT,"Untermen}{2 SPACE}>SIC
     HERN <":PRINT,STRING$(21,"-")          [4B44]
4380 PRINT:PRINT,"Mit welcher Geschwindi
     gkeit sichern?"                        [87D6]
4390 PRINT:PRINT,">1<{2 SPACE}1000 Baud
     (SPEED WRITE 0)"                       [8AB2]
4400 PRINT:PRINT,">2<{2 SPACE}2000 Baud
     (SPEED WRITE 1)"                       [59A8]
4410 PRINT:PRINT,">3<{2 SPACE}3600 Baud
     (TURBO SAVE)"                          [8D96]
4420 f$=e$:GOSUB 620:IF p<1 THEN 2200 EL
     SE IF p>3 THEN 4420                    [A3AE]
4430 IF p=1 THEN SPEED WRITE 0              [BEA6]
4440 IF p=2 THEN SPEED WRITE 1              [80AC]
4450 IF p=3 THEN POKE &B8D1,2:POKE &B8D2
     ,23                                    [76BA]
4460 CLS:f$="Datei{2 SPACE}* "+n$+" *{2
     SPACE}wird gesichert!"+i$:GOSUB 550
                                            [D73A]
4470 PRINT:OPENOUT n$                       [1A24]
4480 PRINT#9,n$:PRINT#9,da$:PRINT#9,s,z1
     ,lz                                    [CAF4]
4490 FOR s1=1 TO s:PRINT#9,b$(s1):NEXT      [7846]
4500 FOR z2=0 TO 10:PRINT#9,dl(z2,1),dl(
     z2,2):NEXT                             [E62C]
4510 PRINT#9,b1,b2,b3,b4,b5                 [9C30]
4520 IF s>5 THEN PRINT#9,b6                 [6938]
4530 IF s>6 THEN PRINT#9,b7                 [6D3E]
4540 IF s>7 THEN PRINT#9,b8                 [7144]
4550 IF s>8 THEN PRINT#9,b9                 [B54A]
4560 IF s>9 THEN PRINT#9,b10                [2D9E]
4570 IF s>10 THEN PRINT#9,b11               [EFF2]
4580 IF s>11 THEN PRINT#9,b12               [DEF8]
4590 IF s>12 THEN PRINT#9,b13               [25FE]
4600 IF s>13 THEN PRINT#9,b14               [AFF2]
4610 IF s>14 THEN PRINT#9,b15               [ACF8]
4620 FOR z=1 TO z1:GOSUB 310                [80AE]
4630 PRINT#9,d1$:PRINT#9,d2$:PRINT#9,d3$
     :PRINT#9,d4$:PRINT#9,d5$               [4BCA]
4640 IF s>5 THEN PRINT#9,d6$                [408A]
4650 IF s>6 THEN PRINT#9,d7$                [3790]
4660 IF s>7 THEN PRINT#9,d8$                [4E96]
4670 IF s>8 THEN PRINT#9,d9$                [259C]
4680 IF s>9 THEN PRINT#9,d10$               [11F0]
4690 IF s>10 THEN PRINT#9,d11$              [6444]
4700 IF s>11 THEN PRINT#9,d12$              [B238]
4710 IF s>12 THEN PRINT#9,d13$              [D03E]
4720 IF s>13 THEN PRINT#9,d14$              [B644]
4730 IF s>14 THEN PRINT#9,d15$              [7C4A]
4740 NEXT:CLOSEOUT:ds=0:PRINT:PRINT"Date
     i ist gesichert!"                      [EDF8]
4750 f$="Bitte Cassette f}r CAT-Ablauf z
     ur}ckspulen!":GOSUB 550:PRINT CHR$(
     7):FOR t=1 TO 4000:NEXT                [37BE]
4760 f$="Cassette f}r CAT-Ablauf zur}ckg
     espult?"+h$:GOSUB 550:GOSUB 710:IF
     t$<>"J" THEN 4760                      [B524]
4770 LOCATE 33,18:PRINT"CAT-Abbruch mit
     >ESC< und irgendeiner Taste"           [3F3E]
4780 f$="Dateisicherung wird }berpr}ft!"
     +i$:GOSUB 550:LOCATE 1,5:CAT           [B52A]
4790 IF p1=9 THEN 4860 ELSE 2200            [FC50]
4800 '                                      [3026]
4810 '***  RAM DISK schlie~en  ***          [E148]
4820 '                                      [EE2A]
4830 f$="RAM DISK wirklich schlie~en?"+h
     $:GOSUB 550:GOSUB 710:IF t$<>"J" TH
     EN 2200 ELSE IF t$="J" AND ds=0 THE
     N 4860                                 [BD6C]
4840 CLS:f$="Daten wurden nicht gesicher
     t!{2 SPACE}Bitte Cassette einlegen!
     ":GOSUB 550:FOR t=1 TO 3000:NEXT       [C47E]
4850 f$="Cassette eingelegt?"+h$:GOSUB 5
     50:GOSUB 710:IF t$="J" THEN p1=9:GO
     TO 4340 ELSE 4850                      [80A4]
4860 f$="RAM DISK ist geschlossen!{2 SPA
     CE}* *{3 SPACE}Zum \ffnen --> |RAMO
     PEN,r:GOTO 10 < eingeben!":GOSUB 55
     0:|RAMCLOSE                            [E244]
4870 CLS:WINDOW 21,60,11,15:PEN 0:PAPER
     1:CLS:LOCATE 11,3:PRINT"* *{2 SPACE
     }E N D E{2 SPACE}* *"                  [20D4]
4880 CALL &BB18:CLS:LOCATE 12,5:PRINT"|R
     AMOPEN,r:GOTO 10":PRINT CHR$(30)       [40F8]
4890 END                                    [FB98]
```

Listing 4. »Futter« für Bank 0 (Schluß)

Das Super-Disketten-Tool

Diskettenmonitor-Programme gibt es viele. Aber nur eins ist so vielseitig wie »Superutility«.

Eigentlich benötigt man zur umfassenden Handhabung seiner Diskettendaten ständig mehrere Utility-Programme: Ein Disketteneditor beugt oft dem Verlust wichtiger Daten vor, dient aber auch vielen anderen Zwecken. Und ein komfortables Werkzeug zur dateiweisen Bearbeitung des Disketteninhalts ist ohnehin wünschenswert. Also – warum nicht alles zu einem leicht zu handhabenden Software-Paket schnüren? Aufgrund dieser Überlegung entstand das Programm »Superutility«, dessen Listing Sie hier finden. Eine Besonderheit kennzeichnet es darüber hinaus: Es ist komplett in der beliebten Sprache Turbo-Pascal geschrieben.

Superutility besteht aus drei einzelnen Programmen, die Sie einzeln oder gemeinsam nutzen können. Aus Platzgründen finden Sie in dieser Ausgabe zunächst den Datei-Editor, der selbständig lauffähig ist. Die Listings der anderen beiden Bestandteile des Paketes folgen im nächsten Schneider-Sonderheft. »Fileedit« arbeitet unter allen CP/M-Versionen, wenn es dort compiliert wurde. Aufgrund des begrenzten Speicherplatzes unter dem Standard CP/M 2.2 der Schneider-Computer, aber auch weil die einzelnen Programmteile auf gemeinsame Unterroutinen zugreifen, verteilt sich das Programm auf insgesamt 20 Listings, die zum Großteil aus Include-Dateien bestehen. Und da man das Rad eigentlich nicht ein zweites Mal zu erfinden braucht, sind viele davon der Turbo-Lader-Programmbibliothek entnommen. Die Veröffentlichung dieser Routinen erfolgt mit freundlicher Genehmigung des Markt&Technik-Buchverlags.

Fileedit ist ein komfortabler Disketteneditor. Sein besonderer Bedienungskomfort ergibt sich aus dem Umstand, daß er nicht sektor- sondern dateiorientiert arbeitet. Der Benutzer muß sich also nicht mühselig durch die Sektorstrukturen der Diskette »hangeln«, um die mitunter arg verstreuten Teile einer Datei zu finden. Vielmehr sucht Fileedit automatisch die Bestandteile der Datei des vorgegebenen Namens zusammen und erlaubt ihre einfache Veränderung. Nebenbei bietet es menügesteuert diverse Dateioperationen wie das Löschen, Umbenennen und Neuanlegen einzelner Dateien.

Bei der Eingabe der Listings müssen Sie darauf achten, die einzelnen Teile unter dem richtigen Dateinamen zu speichern (steht jeweils unter dem Listing). Da Turbo-Pascal während der Compilierung auf die Routinen des entsprechenden Programms zugreift, müssen Sie die richtigen Teile gemeinsam auf die Compiler-Diskette kopieren.

Auf die Handhabung brauchen wir dank der Bildschirmorientierten Benutzerführung nicht näher einzugehen. Nur ein Punkt ist hier erwähnenswert: Ist in den Bedienungshinweisen auf dem Bildschirm die Rede davon, <ESC> zu drücken, ist damit natürlich (CP/M-typisch) die Tastenkombination <CTRL+[> gemeint. (Thomas Bullinger/ja)

Steckbrief	
Programm:	Superutility
Computer:	CPC 464/664/6128
CP/M:	Plus
Datenträger:	Kassette, Diskette
Besonderes:	Turbo-Pascal

```
program  File_editor ;
const max_WA = 8 ;
      max_RA = 1 ;
      max_WS = 30 ;
{$I DECL.UTL}
{$I HEXBIT.UTL}
{$I STRINGS.UTL}
{$I MENU.UTL}
{$I CPM.UTL}
{$I FEN00.UTL}
{$I FEN01.UTL}
{$I FEN02.UTL}
{$I FE00.UTL}
{$I FDEDIT.UTL}
{$I FE01.UTL}
{$I FE02.UTL}
{$I FE03.UTL}
{$I FDSEC00.UTL}
{$I FDSEC01.UTL}
{$I FDSEC02.UTL}
{$I FDSEC03.UTL}
{$I FE04.UTL}
{$I FE05.UTL}

begin
  init ;
  repeat
    index := menu1 (menu_haupt, ' FILEEDIT
                   Version 1.3 ', max_WA) ;
    case (index) of
      1 : ;
      2 : ;
      3 : exec_file ;
      4 : kill_file ;
      5 : ren_file ;
      6 : directory ;
      7 : create_file ;
      8 : sector_file ;
    end ;
  until (index < 3) ;
  des_init ;
  if (index = 2)
  then
  begin
    assign (datei, 'A:SUPERUTL.CHN') ;
    chain (datei) ;
  end
  else
    clrscr ;
end.
```

Listing 1. FILEEDIT.PAS

```
{ Deklarationen fuer ".UTL" - Dateien}
type Work_String       = string [max_WS] ;
     Work_String_array = array [1..max_WA]
                         of Work_String ;
     Real_Array        = array [1..max_RA]
                         of real ;
```

Listing 2. DECL.UTL

```
function hex8
        (bite : byte) : Work_string ;
const  hex_array : array [0..15] of char
                 = '0123456789ABCDEF' ;
begin
  hex8 := hex_array[bite shr 4]
        + hex_array[bite and $0F] ;
end ;

function hex16
        (word : integer) : Work_String ;
const  hex_array : array [0..15] of char
                 = '0123456789ABCDEF' ;
begin
  hex16 := hex8 (word shr 8)
         + hex8 (word and $FF) ;
end ;

function Hex_Dec
        (hex : Work_string) : byte ;

  function  nibble_convert
           (nibble : Work_String) : byte ;
  begin
    nibble := upcase (nibble) ;
    if (ord(nibble) > $39) then
      nibble_convert := ord(nibble) - 55
    else
      nibble_convert := ord(nibble) - $30 ;
  end ;
begin
  if (length (hex) = 2)
  then
  begin
    hex[1] := upcase (hex[1]) ;
    hex[2] := upcase (hex[2]) ;
    Hex_dec :=
    (nibble_convert (hex[1]) * 16)
    + (nibble_convert (hex[2])) ;
  end ;
end ;

function Hex_Dec16
        (hex: Work_string) : integer ;
begin
  if (length (hex) = 4)
  then
  begin
    hex[1] := upcase (hex[1]) ;
    hex[2] := upcase (hex[2]) ;
    hex[3] := upcase (hex[3]) ;
    hex[4] := upcase (hex[4]) ;
    Hex_Dec16 :=
    (Hex_Dec (copy (hex, 1, 2)) * 256)
    + Hex_Dec (copy (hex, 3, 2)) ;
  end ;
end ;

function clrbit
        (word: integer; n: byte): integer ;
begin
  clrbit := word ;
  if (n > 15)
  then
    n := 15 ;
  clrbit := word and not (1 shl n) ;
end ;

function setbit
        (word: integer; n: byte): integer ;
begin
  setbit := word ;
  if (n > 15)
  then
    n := 15 ;
  setbit := word or (1 shl n) ;
end ;

function bitset
        (word: integer; n: byte): boolean ;
begin
  bitset := false ;
  if (n > 15)
  then
    n := 15 ;
 bitset := (word and (1 shl n)) = 1 shl n ;
end ;
```

Listing 3. HEXBIT.UTL

```
var k : integer ;
begin
  search_string := -1 ;
  for k := i to j do
  if (pos(search, search_array[k]) <> 0)
  then search_string := k ;
end ;

procedure replace (var source :
                 Work_string ; from,repl :
                 Work_string ;
                 var count : byte) ;
var position : byte ;
begin
  position := pos(from,source) ;
  while (position <> 0) do
  begin
    delete (source, position,
            length(from)) ;
    insert (repl, source, position) ;
    if (count > 1) then
    begin
      count := count - 1 ;
      position := pos(from,source) ;
    end
    else
      position := 0 ;
  end ;
end ;

function cut
        (var st : Work_string ; delim :
           Work_string) : Work_string ;
var position : byte ;
begin
  cut := '' ;
  position := pos(delim, st) ;
  if (position <> 0) then
  begin
    cut := copy(st, 1, position-1) ;
    st  := copy(st, position+length(delim),
           length(st)) ;
  end ;
end ;
```

Listing 4. STRINGS.UTL

```
function left_string
        (st : Work_string ; tu :
        byte) : Work_string ;
begin
  if (tu < length(st)) then
    left_string := copy(st, 1, tu)
  else
    left_string := st ;
end ;

function mid_string
        (st : Work_string ; from, len :
         byte) : Work_string ;
begin
  mid_string := copy(st, from, len) ;
end ;

function right_string
        (st : Work_string ; num :
        byte) : Work_string ;
begin
  if (num < length(st)) then
    right_string := copy(st,
    length(st)-num+1, num)
  else
    right_string := st ;
end ;

function instr
        (st : Work_String ; search :
        Work_String) : byte ;
var     gefunden : boolean ;
        i        : byte ;
begin
  if ((length(st) < 255)
  and (length(search) < 255))
  then
  begin
    i := 1 ;
    gefunden := false ;
    while ((i <= (length(st)
          - length(search) + 1))
    and (not gefunden)) do
    begin
      if (copy (st, i, length(search)) =
         search)
      then
        gefunden := true
      else
        i := i + 1 ;
    end ;
  end ;
  if (gefunden)
  then
    instr := i
  else
    instr := 0 ;
end ;
```

Listing 4. STRINGS.UTL (Schluß)

```
function menu1
        (menupos : Work_String_array ;
         title   : Work_String ;
         n       : byte) : integer ;
var x,y,i,delta : byte ;
    ch          : char ;
begin
  menu1 := -1 ;
  if (n > 20)
  then
    writeln ('<menu> :
             Zu viele Positionen')
  else
  begin
    if (n > 10)
    then
      delta := 1
    else
      delta := 2 ;
    x := (80 - length (title)) DIV 2 ;
    y := 1 ;
    clrscr ;
    gotoxy (x,y) ;
    highvideo ;
    write (title) ;
      gotoxy (6, 24) ;
      write ('Auswahl durch <^X>, <^E>
              oder Buchstaben, dann
              <RETURN>') ;
      lowvideo ;
      y := 7 - n DIV (2 * (3 - delta)) ;
      x := 25 ;
      for i := 1 to n do
      begin
        gotoxy (x - 6, y + i * delta) ;
        write (chr(64 + i):2, ' -> ',
               menupos[i]) ;
      end ;
      i := 1 ;
      gotoxy (x, y + delta) ;
      highvideo ;
      write (menupos[i]) ;
      lowvideo ;
      ch := ' ' ;
      while (ch <> ^M) do
      begin
        read (kbd, ch) ;
        ch := upcase (ch) ;
        if (ch = ^E) then
        begin
          gotoxy (x, y + i * delta) ;
          write (menupos[i]) ;
          i := i - 1 ;
          if (i <= 0)
          then
            i := n ;
          gotoxy (x, y + i * delta) ;
          highvideo ;
          write (menupos[i]) ;
          lowvideo ;
        end ;
        if (ch = ^X) then
        begin
          gotoxy (x, y + i * delta) ;
          write (menupos[i]) ;
          i := i + 1 ;
          if (i > n)
          then
            i := 1 ;
          gotoxy (x, y + i * delta) ;
```

```
  highvideo ;
  write (menupos[i]) ;
  lowvideo ;
end ;
if ((ord(ch) - 64) in [1..n]) and
   ((ord(ch) - 64) <> i)
then
begin
  gotoxy (x, y + i * delta) ;
  write (menupos[i]) ;
  i := ord(ch) - 64 ;
  gotoxy (x, y + i * delta) ;
  highvideo ;
  write (menupos[i]) ;
          lowvideo ;
        end ;
      end ;
      menu1 := i ;
    end ;
end ;

procedure set_menu_item
         (var menupos : Work_String_array ;
                txt    : Work_String ;
                n      : byte) ;
begin
  menupos[n] := txt ;
end ;
```

Listing 5. MENU.UTL

```
{ Folgende type- und var- Statements
duerfen nicht modifiziert werden: }

type FCBType_TL       =   record
     drive    : byte ;
     name     : array [1..8] of char ;
     typ      : array [1..3] of char ;
     curr_ext : byte ;
     res1     : byte ;
     res2     : byte ;
     rec_cnt  : byte ;
     fab      : array [1..16] of byte ;
end ;
var fcb1 : FCBType_TL ;
    fcb2 : array [0..3] of FCBType_TL ;

procedure set_dma_address (adr : integer) ;
begin
  bdos ($1A,adr) ;
end ;
function first_Dir_entry (drv : byte; var
        search : Work_String) : boolean ;
var i      : byte ;
    result : integer ;
begin
  set_dma_address (addr (fcb2)) ;
  fcb1.drive := drv ;
  move (search[1], fcb1.name, 11) ;
  fillchar (fcb1.curr_ext, 23, 0) ;
  result := bdos ($11, addr(fcb1)) ;
  if (result <> 255) then
  begin
    move (mem[addr(fcb2[result])+1],
          search[1], 11) ;
    search[0] := #11 ;
    for i := 1 to 11 do
      search[i] := chr(ord(search[i])
                   and $7F) ;
    first_dir_entry := true ;
  end
  else
    first_dir_entry := false ;
end ;

function next_dir_entry (var entry :
         Work_string) : boolean ;
var i      : byte ;
    result : integer ;
begin
  set_dma_address (addr (fcb2)) ;
  result := bdos ($12, addr(fcb1)) ;
  if (result <> 255) then
  begin
    move (mem[addr(fcb2[result])+1],
          entry[1], 11) ;
    entry[0] := #11 ;
    for i := 1 to 11 do
      entry[i] := chr(ord(entry[i])
                  and $7F) ;
    next_dir_entry := true ;
  end
  else
    next_dir_entry := false ;
end ;

function login_vector : integer ;
begin
  login_vector := bdosHL ($18) ;
end ;

function disk_param : integer ;
begin
  disk_param := bdosHL ($1F) ;
end ;

function  current_disk : byte ;
begin
  current_disk := bdos ($19) ;
end ;

procedure set_current_disk (drv : byte) ;
begin
  bdos ($0E, drv) ;
end ;

function  current_user : byte ;
begin
  current_user := bdos ($20, 255) ;
end ;

procedure set_current_user (n : byte) ;
begin
  if (n < 16) then
    bdos ($20, n) ;
end ;

function  io_byte : byte ;
begin
  io_byte := bdos (7) ;
```

Listing 6. CPM.UTL

```
end ;

procedure set_io_byte (io_value : byte) ;
begin
  bdos (8, io_value) ;
end ;

function  cpm_version : integer ;
begin
  cpm_version := bdosHL ($0C) ;
end ;

function bios_plus
        (funktion,Akku,RegBC,RegDE,RegHL:
         integer): integer ;
var           BiosPB : record
    func, AReg       : byte ;
    BCreg,DEreg,HLreg: integer ;
end ;
begin
  with BiosPB do
  begin
    func := funktion ;
    AReg := Akku ;
    BCreg:= RegBC ;
    DEreg:= RegDE ;
    HLreg:= RegHL ;
  end;
  if (funktion in [9,16,20,22,25])
  then
    bios_plus := BdosHL(50,Addr(BiosPB))
  else
    bios_plus := Bdos(50,Addr(BiosPB)) ;
end;
```

Listing 6. CPM.UTL (Schluß)

```
type fenster_info =  record
     geschlossen : boolean ;
     ursprung_x  : byte ;
     ursprung_y  : byte ;
     breite      : byte ;
     hoehe       : byte ;
end ;
     modus_type = (normal, invers) ;
const f_info : array [1..10] of
               fenster_info =
               (( geschlossen: true;
               ursprung_x : 1; ursprung_y:
               1; breite: 0; hoehe: 0 ),
               ( geschlossen: true;
               ursprung_x : 1; ursprung_y:
               1; breite: 0; hoehe: 0 ),
               ( geschlossen: true;
               ursprung_x : 1; ursprung_y:
               1; breite: 0; hoehe: 0 ),
               ( geschlossen: true;
               ursprung_x : 1; ursprung_y:
               1; breite: 0; hoehe: 0 ),
               ( geschlossen: true;
               ursprung_x : 1; ursprung_y:
               1; breite: 0;
               hoehe: 0 ), ( geschlossen:
               true; ursprung_x : 1;
               ursprung_y: 1; breite: 0;
               hoehe: 0 ), ( geschlossen:
               true; ursprung_x : 1;
               ursprung_y:  1; breite: 0;
               hoehe: 0 ), ( geschlossen:
               true; ursprung_x : 1;
               ursprung_y: 1; breite: 0;
               hoehe: 0 ), ( geschlossen:
               true; ursprung_x : 1;
               ursprung_y:  1; breite: 0;
               hoehe: 0 ), ( geschlossen:
               true; ursprung_x : 1;
               ursprung_y: 1; breite: 0;
               hoehe: 0 )) ;
      f_kein_Fehler      = 0 ;
      f_falsches_Fenster = 1 ;
      f_schon_offen      = 2 ;
      f_Fenster_zu       = 3 ;
      f_falsches_x       = 4 ;
      f_falsches_y       = 5 ;
      f_falsche_breite   = 6 ;
      f_falsche_hoehe    = 7 ;
      f_Fehler : byte = f_kein_Fehler ;
      leerzeile : string [80] = '

                 ' ;
function f_nummer_ok
         (fensternummer: byte) : boolean ;
begin
  if (fensternummer in [1..10]) then
    f_nummer_ok := true
  else
    f_nummer_ok := false ;
end ;

function f_geschlossen
         (fensternummer: byte) : boolean ;
begin
  if (f_info[fensternummer].geschlossen)
  then
    f_geschlossen := true
  else
    f_geschlossen := false ;
end ;

function f_fehlerstring : Work_string ;
begin
  case f_fehler of
    f_kein_Fehler      :
    f_fehlerstring := 'Kein Fehler' ;
    f_falsches_Fenster :
    f_fehlerstring := 'Falsches Fenster' ;
    f_schon_offen      :
  f_fehlerstring := 'Fenster schon offen' ;
    f_Fenster_zu       :
  f_fehlerstring := 'Fenster geschlossen' ;
    f_falsches_x       :
    f_fehlerstring := 'Falsche x-
                       Koordinate' ;
    f_falsches_y       :
    f_fehlerstring := 'Falsche y-
                       Koordinate' ;
    f_falsche_breite   :
    f_fehlerstring := 'Breite zu gross' ;
    f_falsche_hoehe    :
    f_fehlerstring := 'Hoehe zu gross'
  else f_fehlerstring := 'Unbekannter
                          Fehler' ;
  end ;
end ;
```

Listing 7. FEN00.UTL

```
procedure f_locate
        (fensternummer: byte ;
         px            : byte ;
         py            : byte ;
         modus         : modus_type ) ;
var x, y : byte ;
begin
  f_fehler := f_kein_Fehler ;
  if (f_nummer_ok (fensternummer)) then
  begin
    if (f_geschlossen (fensternummer)) then
      f_fehler := f_Fenster_zu
    else
    begin
      if (px <=
          f_info[fensternummer].breite)
      then
        x :=
           f_info[fensternummer].ursprung_x
           + px - 1
      else
        f_fehler := f_falsches_x ;
      if (py <=
          f_info[fensternummer].hoehe)
      then
        y :=
           f_info[fensternummer].ursprung_y
           + py - 1
      else
        f_fehler := f_falsches_y ;
      if (f_fehler = 0) then
      begin
        gotoxy (x, y) ;
        if (modus = invers) then
          highvideo
        else
          lowvideo ;
      end ;
    end ;
  end
  else
    f_fehler := f_falsches_Fenster ;
end ;
```

Listing 8. FEN01.UTL

```
procedure f_cll
        (fensternummer: byte ;
         py            : byte ;
         modus         : modus_type ) ;
var blank_string : string[80] ;
    px           : byte ;
begin
  f_fehler := f_kein_Fehler ;
  if (f_nummer_ok (fensternummer)) then
  begin
    if (f_geschlossen (fensternummer))
    then
      f_fehler := f_Fenster_zu
    else
    begin
      px := 1 ;
      f_locate (fensternummer, px, py,
                modus) ;
      if (modus = invers) then
        highvideo
      else
        lowvideo ;
      blank_string := copy (leerzeile, 1,
          f_info[fensternummer].breite) ;
      write (blank_string) ;
      lowvideo ;
    end ;
  end
  else
    f_fehler := f_falsches_Fenster ;
end ;

procedure f_clw
        (fensternummer: byte ;
         modus         : modus_type ) ;
var index : byte ;
    py    : byte ;
begin
  f_fehler := f_kein_Fehler ;
  if (f_nummer_ok (fensternummer)) then
  begin
    if (f_geschlossen (fensternummer))
    then
      f_fehler := f_Fenster_zu
    else
    begin
      for index := 1 to
      f_info[fensternummer].hoehe do
      begin
        py := index ;
        f_cll (fensternummer, py, modus) ;
      end ;
    end ;
  end
  else
    f_fehler := f_falsches_Fenster ;
end ;
procedure f_open
  (fensternummer: byte ;
   px            : byte ;
   py            : byte ;
   breite        : byte ;
   hoehe         : byte ) ;
begin
  f_fehler := f_kein_Fehler ;
  if  (f_nummer_ok (fensternummer)) then
  begin
    if (f_geschlossen (fensternummer)) then
    begin
      if (px > 80) then
        f_fehler := f_falsches_x ;
      if (py > 24) then
          f_fehler := f_falsches_y ;
      if (breite > (81 - px)) then
            f_fehler := f_falsche_Breite ;
      if (hoehe > (25 - py)) then
              f_fehler := f_falsche_Hoehe ;
      if (f_fehler = 0) then
      begin
        f_info[fensternummer].geschlossen
        := false ;
        f_info[fensternummer].ursprung_x
```

Listing 9. FEN02.UTL

```
        := px ;
        f_info[fensternummer].ursprung_y
        := py ;
        f_info[fensternummer].breite
        := breite ;
        f_info[fensternummer].hoehe
        := hoehe ;
      end ;
    end
    else
      f_fehler := f_schon_offen ;
  end
  else
    f_fehler := f_falsches_Fenster ;
end ;

procedure f_close (fensternummer : byte ) ;
begin
  f_info[fensternummer].geschlossen
  := true ;
end ;
```

Listing 9. FEN02.UTL (Schluß)

```
type fehlercode = (keine_datei,
                   kein_rename, datei_da) ;
     byte_feld  = array [0..127] of byte ;
const menu_haupt : Work_String_array =
        (    ' Ende von FILEDIT', '
 SUPERUTL aufrufen',
             ' Datei ausfuehren' , '
 Datei loeschen',
             ' Datei umbenennen', '
 Directory',
             ' Datei neu anlegen', '
 Datei bearbeiten') ;
  fen_infos      = 1 ;
  fen_darstellen = 2 ;
  fen_menu       = 3 ;
  fen_eingaben   = 4 ;
  fen_warten     = 5 ;
  fen_fehler     = 6 ;
  sector_nummer  : integer = 0 ;
  sector_modus   : (asc_modus, hex_modus) =
                   asc_modus ;
  sector_index   : byte = 0 ;
var index        : byte ;
    datei        : file ;
    dateiname    : Work_String ;
    sector       : byte_feld ;
procedure fehler
          (fehler_code: fehlercode) ;
begin
  f_clw (fen_infos, normal) ;
  f_clw (fen_eingaben, normal) ;
  f_clw (fen_fehler, normal) ;
  f_locate (fen_fehler, 1, 1, invers) ;
  case (fehler_code) of
    keine_datei : write (' Datei ',
 dateiname, ' nicht vorhanden') ;
    kein_rename : write (' Umbennen in
 ', dateiname, ' unmoeglich') ;
    datei_da    : write (' Datei ',
 dateiname, ' schon vorhanden') ;
  end ;
  lowvideo ;
end ;
```

Listing 10. FE00.UTL

```
procedure warten ;
var eingabe : char ;
begin
  repeat
    eingabe := 'N' ;
    f_cll (fen_warten, 1, normal) ;
    f_locate (fen_warten, 1, 1, invers) ;
    write (' Bitte Leertaste druecken') ;
    read (kbd, eingabe) ;
  until (eingabe = ' ') ;
  lowvideo ;
end ;

procedure init ;
begin
  f_open (fen_infos, 1, 3, 80, 2) ;
  f_open (fen_darstellen, 1, 5, 80, 10) ;
  f_open (fen_menu, 1, 15, 80, 2) ;
  f_open (fen_eingaben, 1, 17, 80, 3) ;
  f_open (fen_warten, 1, 24, 80, 1) ;
  f_open (fen_fehler, 1, 23, 80, 1) ;
end ;

procedure des_init ;
begin
  f_close (fen_infos) ;
  f_close (fen_darstellen) ;
  f_close (fen_menu) ;
  f_close (fen_eingaben) ;
  f_close (fen_warten) ;
  f_close (fen_fehler) ;
end ;

procedure proc_titel (titel: Work_string) ;
begin
  clrscr ;
  highvideo ;
  titel := ' ' + titel + ' ' ;
  writeln (titel) ;
  lowvideo ;
end ;
```

Listing 11. FDEDIT.UTL

```
function  get_datei : boolean ;
var i          : byte ;
begin
  dateiname := ' ' ;
  f_locate (fen_eingaben, 1, 1, normal) ;
  write (' Dateiname ? ') ;
  readln (dateiname) ;
  if (dateiname = ' ')
  then
  get_datei := false
else
begin
  if (mid_string(dateiname, 2, 1) <> ':')
  then
    dateiname := 'A:' + dateiname ;
  for i := 1 to length (dateiname) do
    dateiname [i] := upcase
                     (dateiname[i]) ;
  assign (datei, dateiname) ;
  {$I-} reset (datei) ; {$I+}
  get_datei := (ioresult = 0) ;
```

```
  end ;
end ;

procedure exec_file ;
begin
  proc_titel ('Datei ausfuehren') ;
  if (get_datei)
  then
  begin
    des_init ;
    clrscr ;
    close (datei) ;
    execute (datei) ;
  end
  else
  begin
    fehler (keine_datei) ;
    warten ;
  end ;
end ;

procedure kill_file ;
var eingabe : char ;
begin
  proc_titel ('Datei loeschen') ;
  if (get_datei)
  then
  begin
    close (datei) ;
    f_clw (fen_eingaben, normal) ;
    f_locate (fen_eingaben, 1, 1, normal) ;
    write (' Datei ', dateiname, ' loeschen
            (j/n) ? ') ;
    eingabe := ' ' ;
    read (kbd, eingabe) ;
    if (eingabe in ['j','J'])
    then
    begin
      erase (datei) ;
      f_clw (fen_infos, normal) ;
      f_clw (fen_eingaben, normal) ;
      f_locate (fen_infos, 1, 1, invers) ;
      writeln (' Die Datei ', dateiname, '
                ist jetzt geloescht ') ;
      lowvideo ;
    end
    else
      f_clw (fen_eingaben, normal) ;
      warten ;
  end
  else
  begin
    fehler (keine_datei) ;
    warten ;
  end ;
end ;
```

Listing 12. FE01.UTL

```
procedure ren_file ;
begin
  proc_titel ('Datei umbennen') ;
  if (get_datei)
  then
  begin
    close (datei) ;
    dateiname := ' ' ;
    f_locate (fen_eingaben, 1, 2, normal) ;
    write (' Neuer Dateiname ? ') ;
    readln (dateiname) ;
    if (dateiname <> '')
    then
    begin
      if (mid_string (dateiname, 2, 1)
        <> ':')
      then
        dateiname := 'A:' + dateiname ;
        {$I-} rename (datei, dateiname) ;
        {$I+}
        if (ioresult <> 0)
        then
        begin
          fehler (kein_rename) ;
        end ;
    end ;
  end ;
  warten ;
end ;

procedure directory ;
var eingabe      : char ;
    end_of_dir   : boolean ;
    laufwerk     : byte ;
begin
  proc_titel ('Directory') ;
  f_locate (fen_eingaben, 1, 1, normal) ;
  write (' Laufwerk (A/B) ? ') ;
  readln (eingabe) ;
  eingabe := upcase (eingabe) ;
  if (eingabe in ['A','B'])
  then
  begin
    f_clw (fen_eingaben, normal) ;
    f_locate (fen_infos, 1, 1, normal) ;
    write (' DIRECTORY fuer Laufwerk ',
             eingabe, ':') ;
    f_locate (fen_darstellen, 1, 1,
               invers) ;
    dateiname := '???????????' ;
    laufwerk := ord (eingabe) - 64 ;
    if (first_dir_entry (laufwerk,
        dateiname))
    then
    begin
      end_of_dir := false ;
      write (dateiname:16) ;
      repeat
        if (next_dir_entry (dateiname))
        then
          write (dateiname:16)
        else
          end_of_dir := true ;
      until (end_of_dir) ;
      lowvideo ;
    end
    else
      write (' Directory leer ') ;
  end
  else
    f_clw (fen_eingaben, normal) ;
  warten ;
end ;
```

Listing 13. FE02.UTL

```
procedure create_file ;
var i, no_sektoren : integer ;
begin
  proc_titel ('Datei neu anlegen') ;
  if (get_datei)
  then
    fehler (datei_da)
  else
  begin
    close (datei) ;
    f_locate (fen_eingaben, 1, 2, normal) ;
    write ('Wieviel Sektoren (0 = Kein
            Schreiben) ? ') ;
    readln (no_sektoren) ;
    if (no_sektoren > 0)
    then
    begin
      rewrite (datei) ;
      for i := 1 to no_sektoren do
      begin
        fillchar (sector, 128, $5A) ;
        blockwrite (datei, sector, 1) ;
      end ;
      close (datei) ;
    end ;
  end ;
  warten ;
end ;
```

Listing 14. FE03.UTL

```
function hex_asc_y (i: byte) : byte ;
begin
  hex_asc_y := 2 + (i div 16) ;
end ;
function hex_x (i: byte) : byte ;
begin
  hex_x := 5 + ((i mod 16) * 3) ;
end ;
function asc_x (i: byte) : byte ;
begin
  asc_x := 55 + (i mod 16) ;
end ;
procedure cursor_rechts ;
begin
  if (sector_index < 127)
  then
    sector_index := sector_index + 1 ;
end ;
procedure cursor_links ;
begin
  if (sector_index > 0)
  then
    sector_index := sector_index - 1 ;
end ;
procedure cursor_hoch ;
begin
  if (sector_index > 15)
  then
    sector_index := sector_index - 16 ;
end ;
procedure cursor_runter ;
begin
  if (sector_index < 112)
  then
    sector_index := sector_index + 16 ;
end ;
```

Listing 15. FDSEC00.UTL

```
procedure asc_editieren ;
var eingabe : char ;
begin
  repeat
    f_locate (fen_darstellen,
              asc_x(sector_index),
              hex_asc_y(sector_index),
               normal) ;
    eingabe := ' ' ;
    read (kbd, eingabe) ;
    if (ord (eingabe) < $20)
    then
    begin
      case (eingabe) of
      ^D : begin
        cursor_rechts ;
        f_locate (fen_darstellen,
                  asc_x(sector_index),
                  hex_asc_y(sector_index),
                  normal) ;
      end ;
      ^S : begin
        cursor_links ;
        f_locate (fen_darstellen,
                  asc_x(sector_index),
                  hex_asc_y(sector_index),
                  normal) ;
      end ;
      ^E : begin
        cursor_hoch ;
        f_locate (fen_darstellen,
                  asc_x(sector_index),
                  hex_asc_y(sector_index),
                  normal) ;
      end ;
      ^X : begin
        cursor_runter ;
        f_locate (fen_darstellen,
                  asc_x(sector_index),
                  hex_asc_y(sector_index),
                  normal) ;
      end ;
    end ;
  end
  else
  begin
    sector[sector_index] := ord(eingabe) ;
    write (eingabe) ;
    f_locate (fen_darstellen,
              hex_x(sector_index),
              hex_asc_y(sector_index),
              normal) ;
    write (hex8 (ord (eingabe)):3) ;
    cursor_rechts ;
    f_locate (fen_darstellen,
              asc_x(sector_index),
              hex_asc_y(sector_index),
              normal) ;
  end ;
  until (eingabe = ^[) ;
end ;
```

Listing 16. FDSEC01.UTL

```
procedure hex_editieren ;
  function hex_eingabe
           (erster_char : char) :
           Work_string ;
  var  hex_char   : char ;
       hex_string : string [2] ;
  begin
    hex_eingabe := '00' ;
    hex_char := upcase (erster_char) ;
    if (hex_char in ['0'..'9', 'A'..'F'])
    then
    begin
      hex_string := hex_char ;
      read (kbd, hex_char) ;
      hex_char := upcase (hex_char) ;
      if (hex_char in ['0'..'9', 'A'..'F'])
      then
        hex_eingabe := hex_string
                     + hex_char ;
    end ;
  end ;
var eingabe  : char ;
    hex_str  : Work_string ;
    hex_zahl : byte ;
begin
  repeat
    f_locate (fen_darstellen,
              hex_x(sector_index) + 1,
              hex_asc_y(sector_index),
              normal) ;
    eingabe := ' ' ;
    read (kbd, eingabe) ;
    if (ord (eingabe) < $20)
    then
    begin
      case (eingabe) of
      ^D : begin
        cursor_rechts ;
        f_locate (fen_darstellen,
                  hex_x(sector_index),
                  hex_asc_y(sector_index),
                  normal) ;
      end ;
      ^S : begin
        cursor_links ;
        f_locate (fen_darstellen,
                  hex_x(sector_index),
                  hex_asc_y(sector_index),
                  normal) ;
      end ;
      ^E : begin
        cursor_hoch ;
        f_locate (fen_darstellen,
                  hex_x(sector_index),
                  hex_asc_y(sector_index),
                  normal) ;
      end ;
      ^X : begin
        cursor_runter ;
        f_locate (fen_darstellen,
                  hex_x(sector_index),
                  hex_asc_y(sector_index),
                  normal) ;
      end ;
    end ;
  end
  else
  begin
    hex_str := hex_eingabe (eingabe) ;
    hex_zahl := hex_dec (hex_str) ;
    sector[sector_index] := hex_zahl ;
    write (hex_str) ;
    f_locate (fen_darstellen,
              asc_x(sector_index),
              hex_asc_y(sector_index),
              normal) ;
    if (hex_zahl in [$20..$7E])
    then
      write (chr (hex_zahl))
    else
      write ('.') ;
    cursor_rechts ;
    f_locate (fen_darstellen,
              hex_x(sector_index),
              hex_asc_y(sector_index),
              normal) ;
  end ;
  until (eingabe = ^[) ;
end ;
```

Listing 17. FDSEC02.UTL

```
procedure sector_editieren ;
begin
  f_clw (fen_menu, normal) ;
  f_locate (fen_menu, 1, 1, normal) ;
  write ('^S - Links, ^D - Rechts,
         ^E - Hoch, ^X - Runter, ') ;
  write ('ESC - Quit (Ende)') ;
  sector_index := 0 ;
  case (sector_modus) of
    asc_modus : asc_editieren ;
    hex_modus : hex_editieren ;
  end ;
end ;
procedure sector_darstellen ;
var i : byte ;
begin
  f_locate (fen_darstellen, 1, 1, normal) ;
  case (sector_modus) of
    asc_modus : write (' ASC ') ;
    hex_modus : write (' HEX ') ;
  end ;
  f_locate (fen_darstellen, 6, 1, invers) ;
  for i := 0 to 15 do
    write (hex8 (i):2, ' ') ;
  lowvideo ;
  for i := 0 to 127 do
  begin
    if (i mod 16 = 0)
    then
    begin
      f_locate (fen_darstellen, 1,
```

Listing 18. FDSEC03.UTL

```
            hex_asc_y(i), invers) ;
    write (hex8 (i):3, ':') ;
    lowvideo ;
  end ;
  f_locate (fen_darstellen, hex_x(i),
            hex_asc_y(i), normal) ;
  write (hex8 (sector[i]):3) ;
  f_locate (fen_darstellen, asc_x(i),
            hex_asc_y(i), normal) ;
    if (sector[i] in [$20..$7F])
    then
      write (chr (sector[i]))
    else
      write ('.') ;
  end ;
end ;
```

Listing 18. FDSE03.UTL (Schluß)

```
procedure f_sector_nummer_holen ;
var sector_no_neu : integer ;
begin
  f_clw (fen_eingaben, normal) ;
  f_locate (fen_eingaben, 2, 1, normal) ;
  write ('Sektornummer (0 - ',
          filesize(datei)-1, ') ? ') ;
  readln (sector_no_neu) ;
  if (sector_no_neu < 0)
  then
    sector_nummer := filesize(datei) - 1
  else
  begin
    if (sector_no_neu >
       (filesize(datei)-1))
    then
      sector_nummer := 0
    else
      sector_nummer := sector_no_neu ;
  end ;
end ;

procedure f_letzter_sector
          (sector_no_alt: integer) ;
begin
  if (sector_no_alt > 0)
  then
    sector_nummer := sector_no_alt - 1
  else
    sector_nummer := filesize(datei) - 1 ;
end ;
procedure f_naechster_sector
          (sector_no_alt: integer) ;
begin
  if (sector_no_alt < (filesize(datei)-1))
  then
    sector_nummer := sector_no_alt + 1
  else
    sector_nummer := 0 ;
end ;

procedure find_string ;
var eingabe    : Work_String ;
    such_strg  : Work_String ;
    such_index : integer ;
    val_fehler : integer ;
  function  gleichheit
        (org_feld : byte_feld) : boolean ;
  var    such_feld : array [0..max_WS] of
                  byte absolute such_strg ;
         gleich    : boolean ;
         ig        : byte ;
  begin
    gleich := true ;
    ig     := 0 ;
    while ((gleich) and (ig
         < length(such_strg))) do
    begin
      gleich := (org_feld[ig]
             = such_feld[ig+1]) ;
      ig := ig + 1
    end ;
    gleichheit := gleich ;
  end ;

function insec
        (org_sec : byte_feld) : integer ;
var such_sec : array [0..max_WS] of byte
               absolute such_strg ;
    gefunden : boolean ;
    ii       : byte ;
    hilfs_sec: byte_feld ;
begin
  ii := 0 ;
  gefunden := false ;
  while ((ii <= (sizeof(org_sec)
        - length(such_strg)))
  and (not gefunden)) do
  begin
    move (org_sec[ii], hilfs_sec,
          length(such_strg)) ;
    gefunden := gleichheit (hilfs_sec) ;
    ii := ii + 1 ;
  end ;
  if (gefunden)
  then
    insec := ii - 1
  else
    insec := -1 ;
end ;


procedure string_suchen ;
var gefunden   : boolean ;
    find_index : integer ;
begin
  gefunden := false ;
  while ((such_index < filesize(datei))
  and (not gefunden)) do
  begin
    seek (datei, such_index) ;
    blockread (datei, sector, 1) ;
    find_index := insec (sector) ;
    if (find_index <> -1)
    then
    begin
      gefunden := true
    end
    else
      such_index := such_index + 1 ;
  end ;
  if (gefunden)
  then
    sector_nummer := such_index
```

```
    else
    begin
      write (^G^G^G) ;
      seek (datei, sector_nummer) ;
      blockread (datei, sector, 1) ;
    end ;
  end ;
begin
  f_clw (fen_eingaben, normal) ;
  f_locate (fen_eingaben, 1, 1, normal) ;
  write (' Suchstring (max. ', max_WS, '
         Zeichen) ? ') ;
  eingabe := '' ;
  readln (eingabe) ;
  if (eingabe <> '')
  then
  begin
    such_strg := eingabe ;
    f_locate (fen_eingaben, 1, 2, normal) ;
    write (' Ab Sektor (RETURN =
           aktueller) ? ') ;
    eingabe := '' ;
    readln (eingabe) ;
    if (eingabe = '')
    then
    begin
      such_index := sector_nummer ;
      string_suchen ;
    end
    else
    begin
    val (eingabe, such_index, val_fehler) ;
      if (val_fehler = 0)
      then
        string_suchen;
    end ;
  end ;
end ;
```

Listing 19. FE04.UTL

```
procedure sector_bearbeiten ;
var eingabe : char ;
begin
  f_clw (fen_eingaben, normal) ;
  f_locate (fen_infos, 2, 1, normal );
  write (dateiname) ;
  sector_darstellen ;
  repeat
    f_locate (fen_infos, 20, 1, normal) ;
    writeln ('Sektor: ', sector_nummer:5) ;
    f_locate (fen_menu, 1, 1, normal) ;
    write ('A - ASCII, B - Sektornummer,
            E - Editieren, F - Suchen, ') ;
    writeln ('H - Hex,') ;
    write ('L - Letzter Sektor,
            N - Naechster Sektor, ') ;
    writeln ('S - Speichern,
              Q - Quit (Ende)') ;
    f_locate (fen_eingaben, 1, 1, normal) ;
    write (' Befehl ? ') ;
    eingabe := ' ' ;
    readln (eingabe) ;
    eingabe := upcase (eingabe) ;
    f_cll (fen_eingaben, 1, normal) ;
    case (eingabe) of
      'A' : begin
        sector_modus := asc_modus ;
        f_locate (fen_darstellen, 1,
                  1, normal) ;
        write (' ASC ') ;
      end ;
      'B' : begin
        f_sector_nummer_holen ;
        f_cll (fen_eingaben, 1, normal) ;
        seek (datei, sector_nummer) ;
        blockread (datei, sector, 1) ;
        sector_darstellen ;
      end ;
      'E' : begin
        sector_editieren ;
        f_clw (fen_menu, normal) ;
        f_clw (fen_eingaben, normal) ;
      end ;
      'F' : begin
        find_string ;
        sector_darstellen ;
        f_clw (fen_eingaben, normal) ;
      end ;
      'H' : begin
        sector_modus := hex_modus ;
        f_locate (fen_darstellen, 1,
                  1, normal) ;
        write (' HEX ') ;
      end ;
      'L' : begin
        f_letzter_sector (sector_nummer) ;
        seek (datei, sector_nummer) ;
        blockread (datei, sector, 1) ;
        sector_darstellen ;
      end ;
      'N' : begin
       f_naechster_sector (sector_nummer) ;
        seek (datei, sector_nummer) ;
        blockread (datei, sector, 1) ;
        sector_darstellen ;
      end ;
      'S' : begin
        seek (datei, sector_nummer) ;
        blockwrite (datei, sector, 1) ;
      end ;
      'Q' : ;
      else write (^G^G^G) ;
    end ;
    until (eingabe = 'Q') ;
end ;


procedure sector_file ;
begin
  proc_titel ('Datei bearbeiten') ;
  if (get_datei)
  then
  begin
    sector_nummer := 0 ;
    blockread (datei, sector, 1) ;
    sector_bearbeiten ;
  end
  else
    fehler (keine_datei) ;
  close (datei) ;
  warten ;
end ;
```

Listing 20. FE05.UTL

Knifflige Diamantensuche

Im Spiel »Stone Runner« steuern Sie das kleine Männchen »Bity« durch Labyrinthe und versuchen, möglichst viele der dort verstreuten Diamanten einzusammeln. Doch Vorsicht – es lauern viele Gefahren.

Bity hat die Aufgabe, Diamanten zu sammeln. »Kein Problem«, denkt Bity bei sich, »das wär ja wohl gelacht.« So zieht er also los, die Diamanten zu holen. Hätte er gewußt, welch schwierige Aufgabe ihn erwartet, wäre er wohl nicht so leichtfertig gewesen.

Das Gebiet, in dem Bity die Diamanten sammeln soll, sieht recht merkwürdig aus. Überall sind Löcher, in die Bity hineinfallen kann. Steckt er erst mal in einem, kommt er nicht wieder heraus. An anderen Stellen gibt es allerdings auch Hilfen für ihn. So darf er Leitern, Kletterseile und Sprungbretter benutzen, um Hindernisse zu überwinden. Auch hat er eine ätzende Flüssigkeit dabei, so daß er sich gegebenenfalls seinen Weg freiätzt.

Es gibt jedoch noch weitere Probleme, deren Bewältigung Bity nicht gerade leichtfällt. Zum einen hat man ihm nur einen begrenzten Zeitraum zugestanden, um ein Gebiet nach allen Diamanten abzusuchen, zum anderen kostet ihn jeder Schritt Kraft, die er allerdings von Zeit zu Zeit wieder auffrischen kann. Um an die Diamanten zu gelangen, muß er sich jeden Schritt genau überlegen.

Das Programm »Stone Runner« läuft auf allen CPCs und besteht aus vier Listings. Das Basic-Hauptprogramm finden Sie in Listing 1. Geben Sie es bitte mit Explora ein und speichern Sie es unter dem Namen »SR.BAS«. Da die Bildschirmwiedergabe des Stone Runner auf die Verwendung eines Farbmonitors abgestimmt ist, sollten Sie für den Grünmonitor die Farbregister so ändern, wie es dort empfohlen ist. Die Listings 2, 3 und 4 enthalten Binärcodes. Der Maschinencode für zeitkritische Aufgaben (Bildaufbau etc.) steht in Listing 2. Der Bewegung der Spielfigur Bity dienen die Sprite-Routinen aus Listing 3. Beide sind als DATA-Lader wahlweise mit Explora oder CPC einzugeben.

Zum Abtippen des Listings 4 empfiehlt sich unbedingt unsere Eingabehilfe »CPC«, deren Listing und Bedienungsanleitung Sie ab Seite 87 dieser Ausgabe finden.

Doch Sie können sich die Eingabe des sehr langen Listing 4 auch sparen, da es »nur« neun fertige Bilder als Spielfelder enthält. Zu diesem Zweck gehen Sie nach dem Start des Stone Runner mit »RUN "SR"« in dessen Editor (Menüpunkt »Konstruieren«). Die Bilder 1 bis 6 zeigen sechs der genannten neun Spielfelder, damit Sie erst einmal einen Grundstock haben. Geben Sie diese Bilder nacheinander ein und speichern Sie sie (Menüpunkt »Speichern« und dann <B> für »Bilder« wählen) unter einem beliebigen Dateinamen (die Extension »PIC« vergibt der Computer automatisch). Bei der Eingabe des Listings mit CPC gehen Sie wie folgt vor. Starten Sie zunächst CPC. Wählen Sie dann <C> für »Code eingeben«. Die folgende Abfrage der Eingabeparameter überspringen Sie mit <ENTER> (oder <RETURN>). Für »Startnummer« geben Sie »1000« als Zeilennummer der ersten Zeile, für »Schrittweite« als Inkrement eine 1 ein. Nun erwartet CPC die Eingabe der Startadresse. Dabei handelt es sich um die erste vierstellige Hexadezimalzahl am Anfang des Listings (hier 61A0). Bei der Länge des Binärteils bieten sich kleine Arbeitspausen an. Dazu merken Sie sich die Adresse der Zeile hinter der zuletzt eingegebenen und unterbrechen die Eingabe mit <ESC>. Aus dem Hauptmenü wählen Sie dann <S> für »Schreibe Code« und speichern so das Teilergebnis. Wenn Sie später mit der Eingabe fortfahren wollen, laden Sie erst den gespeicherten Teil mit <L> an seine Original-Ladeadresse (Sie erinnern sich: die erste Adresse des Listings, also 61A0 hex). Nun gehen Sie mit <C> wieder in den Eingabemodus, wählen jetzt als Startadresse und Startnummer für die Eingabe aber die Werte,

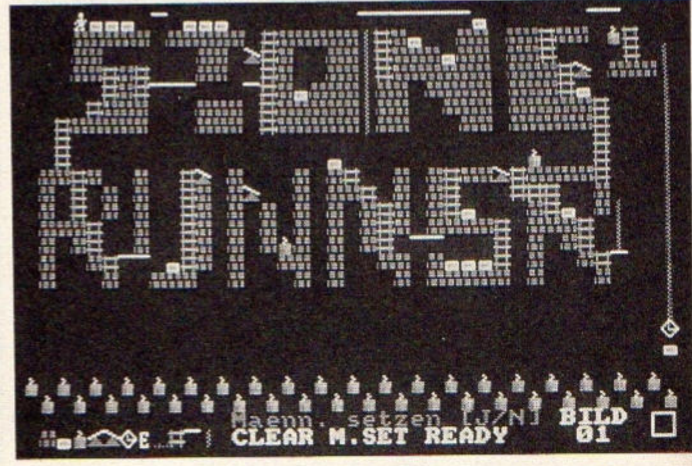


Bild 1. Das erste Bild ist noch relativ leicht zu schaffen

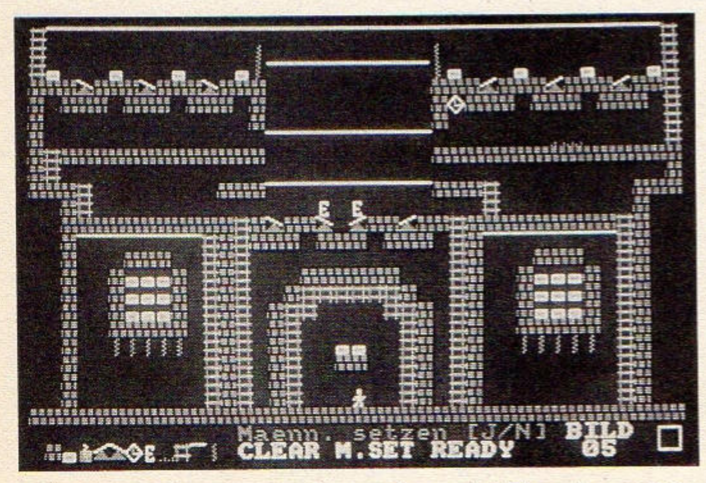


Bild 3. Im fünften Level müssen Sie gezielt Energie sparen

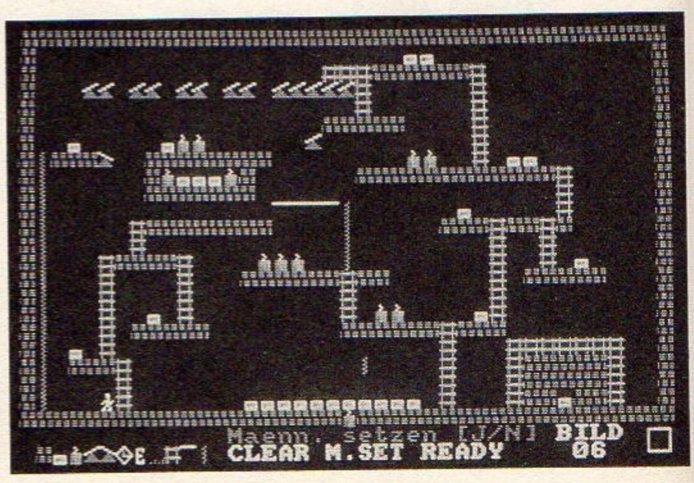


Bild 4. Wer findet den kürzesten Weg?

die Sie sich beim Speichern als Ende gemerkt haben. Dort fahren Sie dann mit der Eingabe fort. Nachdem Sie diese Arbeit erfolgreich hinter sich brachten, speichern Sie den komplett eingegebenen Binärcode (wiederum mit <S>). Dabei fragt Sie der Computer jeweils nach zwei Parametern. Die Startadresse von »SR.PIC« (Listing 4) lautet 61A7 hex. Lassen Sie sich bitte nicht durch die Eingabe-Startadresse 61A0 hex in die Irre führen, 61A7 ist in diesem Fall korrekt. Die Endadresse als zweiter Parameter zum Speichern ist 821C hex.

Nun muß Ihre Diskette mindestens die vier Dateien SR.BAS, SR.BIN, SR.SPR und SR.PIC enthalten (bei Verzicht auf die fertigen Bilder aus SR.PIC natürlich nur drei). Diverse Lade- und Speichervorgänge der Bilder und Bestzeiten machen Stone Runner für Kassettenbetrieb eigentlich ungeeignet. Wollen Sie auf dieses tolle Spiel als Besitzer eines CPC 464 ohne Diskettenlaufwerk trotzdem nicht verzichten (was angesichts seiner Qualitäten durchaus verständlich ist), speichern Sie auf der Kassette zuerst Listing 1. Dahinter muß direkt der Maschinencode aus Listing 2 unter dem Namen »SR.BIN« gespeichert sein. Im Anschluß daran erwartet Stone Runner die Spriteroutinen aus Listing 3 unter dem Dateinamen »SR.SPR«. Als letztes speichern Sie die Levels aus Listing 4 (oder nach der Eingabe im Editor) unter dem Namen »SR.PIC«. Auf die Nutzung der »Hightimes«-Liste sollten Sie im Kassettenbetrieb jedoch verzichten, da Sie sonst durch andauerndes Speichern und Laden kaum noch zum Spielen kommen.

Jetzt ist der freudige Zeitpunkt erreicht, an dem Sie für Ihre Fleißarbeit belohnt werden. Starten Sie also Stone Runner mit »RUN"SR"«. Im Menü bewegen Sie den Cursor (invertierter Balken) mit einem Joystick auf »Laden« und drücken die Feuertaste. Die Frage

```
(B)ilder oder (H)ightimes laden?
```

beantworten Sie mit <B>. Als Dateinamen geben Sie nun »SR« ein. Eine weitere Frage erscheint:

```
Alte Bilder löschen (j/n) ???
```

Mit Ihrer Antwort entscheiden Sie, ob Stone Runner die Bilder anstelle der Bilder im Arbeitsspeicher lädt (<J>) oder sie an vorhandene als weitere Levels anhängt (<N>). Da sich zu diesem Zeitpunkt ja noch keine Bilder im RAM befinden, wählen Sie natürlich <J>. Nach einem kurzen Augenblick erscheint wieder das Menü.

Spielen funktioniert naturgemäß nur, wenn Sie vorher eine Bild-Datei mit dem Menüpunkt »Laden« geladen oder mit dem Editor Bilder konstruiert haben. Bewegen Sie den Joystick, um sich eins der Bilder auszusuchen und drücken Sie zum Spielen die Feuertaste.

Ihnen stehen anfangs sieben Versuche frei, um Bity alle Diamanten einsammeln zu lassen. Erst dann gelangen Sie ins nächste Bild (Level). Meist gibt es nur einen einzigen Weg, das Level zu überwinden. Stürzt Bity ab und fällt dadurch tiefer als sechs »Stockwerke«, oder berührt er eine der herumliegenden Bomben, ist einer der Versuche vertan. Ebenso kostet es eine Chance, wenn Bitys Zeit zu Ende ist. Sie steuern Bity mit einem normalen Joystick nach links und rechts. Nur auf Leitern kann er sich sowohl nach oben als auch nach unten bewegen. Abgründe überwindet er mit Hilfe der waagerechten Kletterstangen; Bity hält sich automatisch daran fest. Er läßt sie erst los, wenn Sie den Joystick nach unten ziehen. Die Steine links und rechts unter sich ätzt er weg, wenn Sie den Feuerknopf festhalten und den Joystick gleichzeitig in die entsprechende Richtung drücken. Das funktioniert indes nur, wenn Bity noch über genug Energie verfügt. Einmaliges Ätzen kostet ihn zehn Energiepunkte. Wenn Bity stürzt, verliert er pro Stockwerk ebenfalls einen Energiepunkt. Ebenso unangenehme Effekte beinhalten die Steine mit Reißnägeln.

Sprungfedern schleudern Bity automatisch in die entsprechende Richtung. An Seilen kann Bity nur nach unten klettern. Punkte erntet Bity durch Sammeln der Diamanten. Hat er alle Diamanten eines Levels bei sich, bekommt er die verbleibenden Zeiteinheiten auf seinem Punktekonto gutgeschrieben. Außer den Diamanten gibt es in einigen Bildern auch einen Energie- und/oder Zeitbonus. Eine solche »Kraftpille« (großes »E«) füllt das Energiekonto um 50 Einheiten auf. Der Zeitbonus (eine stilisierte Uhr) ist um so höher, je geringer die Restzeit ist; es lohnt sich also, etwas Geduld aufzubringen und zu warten. Ihre Eintragungen in die Tabelle der besten Zeiten (»Hightimes«) speichern Sie nach dem Spiel über den Menüpunkt »Speichern«, Unterpunkt »(H)ightimes«. Diese Tabelle wird bei jedem weiteren

Bild 2. Level 2 verlangt nach überlegter Zeiteinteilung

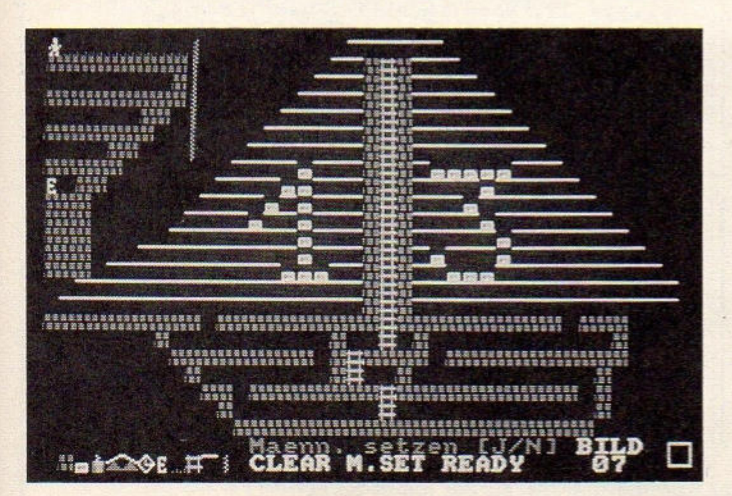


Bild 5. Der Energiebonus allein hilft nicht viel...

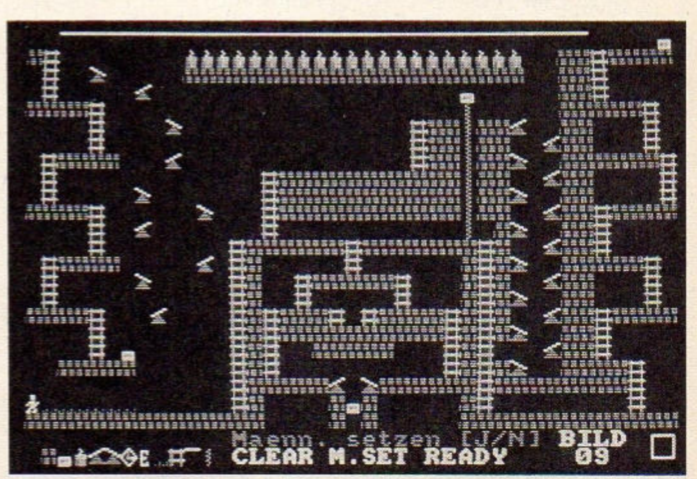


Bild 6. Lassen Sie sich keine grauen Haare wachsen

Spiel fortgeführt, wenn Sie vor dem Beginn des ersten Spiels nach dem Programmstart und dem Laden der Bilder die alten Bestzeiten laden.

Konstruieren. Haben Sie diesen Menüpunkt gewählt, erscheint in der untersten Bildschirmzeile eine Menüleiste, die sämtliche Spielelemente und die Punkte »Clear«, »M. Set« und »Ready« zeigt. Drücken Sie nun den Feuerknopf und Sie sehen ein Fadenkreuz, mit dem Sie eins der Elemente auswählen. Bewegen Sie das Fadenkreuz an die gewünschte Bildschirmposition und drücken Sie wiederum den Feuerknopf, erscheint das Element. Um in das Auswahlmenü zurückzukehren, brauchen Sie bloß die Leertaste zu drücken. Wenn das Bild fertig ist, müssen Sie nur noch Bity hineinsetzen. Dazu positionieren Sie das Fadenkreuz an der gewünschten Stelle, drücken die Leertaste und aktivieren mit dem Joystick den Menüpunkt »M. Set«. Dann wählen Sie den Menüpunkt »Ready«. »Ready« funktioniert nur, wenn das Spielfeld mindestens einen Diamanten enthält und Bitys Position markiert ist.

Editieren. Sie sehen auf dem Monitor das erste Bild. Wählen Sie durch Bewegung des Joysticks ein Bild zur Bearbeitung. Der Unterschied zum Konstruieren liegt darin, daß Sie ein vorhandenes Bild verändern.

Abspeichern. Speichert die Bilder im Arbeitsspeicher auf dem angeschlossenen Datenträger. Als Eingabe des Dateinamen akzeptiert Stone Runner maximal acht Zeichen ohne Punkt. Die Extension ».PIC« hängt der Computer zur Kennzeichnung automatisch an.

Test. Spielen Sie ein editiertes Bild probehalber, um zu sehen, ob es zu meistern ist.

Laden. Lädt gespeicherte Bilder in den Arbeitsspeicher des Computers. Geben Sie auf die Frage nach dem Löschen der alten Bilder <N> ein, werden die neuen Bilder an die schon im Speicher befindlichen angehängt.

(Thomas Stein/hf/ja)

Steckbrief	
Programm:	Stone Runner
Computer:	CPC 464/664/6128
Checksummer:	Explora/CPC
Datenträger:	Diskette, (Kassette)
Besonderes:	Joystick-Steuerungen

```
1000 '****************************          [62EC]
1010 '* S T O N E - R U N N E R *           [351A]
1020 '****************************          [EAF0]
1030 '                                      [9016]
1040 'written by Thomas Stein               [2086]
1050 '                                      [B61A]
1060 IF PEEK(&A000)<>1 THEN SYMBOL AFTER
      256:MEMORY 24998:LOAD"!sr.bin":CAL
     L &A000:LOAD"!sr.spr":POKE 24999,0:
     CLEAR                                  [C376]
1070 '                                      [B41E]
1080 '                                      [B920]
1090 'Variablen & Funktionen                [4858]
1100 '                                      [8F12]
1110 bilder=24999:maxbilder=16:ENV 1,14,
     1,10:ENV 2,14,-1,10:ENV 3,100,-2,2:
     ENT 1,100,-2,2:ENV 4,14,-1,3,14,1,3
     :ENT 2,14,-5,3,14,5,3:ENV 5,14,-1,3
     :ENV 6,7,-1,1,10,0,1:ENT 3,7,5,1:EN
     V 7,7,-1,1,10,0,1:DIM top5$(maxbild
     er,5),top5(maxbilder,5)                [BAF6]
1120 DEF FNadr(x)=bilder+1+(x-1)*923:DEF
      FNpadr(l,x,y)=FNadr(l)+x/2+y/8*40     [33B0]
1130 DEF FNz$(z,l)=STRING$(l-LEN(STR$(z)
     )+1,"0")+MID$(STR$(z),2)               [C1BE]
1140 IF PEEK(6)=128 THEN DEF FNvers$(bil
     d,platz)=DEC$((top5(bild,platz),"##
     .##")+"sec" ELSE DEF FNvers$(bild,p
     latz)=DEC$(top5(bild,platz),"##.##"
     )+"sec"                                [BB4E]
1150 DEF FNl$(l)=STRING$(l,249)+" ":DEF
     FNtop$(bild,platz)=top5$(bild,platz
     )+STRING$(14-LEN(top5$(bild,platz))
     ,".")+CHR$(9)+CHR$(9)+FNvers$(bild,
     platz)                                 [6A82]
1160 DEF FNscreen(x,y)=40000+y*5+x/2        [B722]
1170 DIM um$(3):um$(1)="CLEAR":um$(2)="M
     .SET":um$(3)="READY":FOR i=1 TO max
     bilder:FOR j=1 TO 5:top5$(i,j)="?.?
     .?":top5(i,j)=0:NEXT j,i:name$="ABC
     DEFGHIJKLMNOPQRSTUVWXYZ.!"+CHR$(242
     )+"#"                                  [3038]
1180 DIM m$(7):RESTORE 1180:FOR i=1 TO 7
     :READ m$(i):NEXT:rahmen$=CHR$(150)+
     STRING$(12,154)+CHR$(156):FOR i=1 T
     O 13:rahmen$=rahmen$+CHR$(149)+STRI
     NG$(12,32)+CHR$(149):NEXT:rahmen$=r
     ahmen$+CHR$(147)+STRING$(12,154)+CH
     R$(153):ende$="GAME-OVER"              [5A30]
1190 DATA SPIELEN,KONSTRUIEREN,EDITIEREN
     ,TEST,SPEICHERN,LADEN,ENDE             [27C0]
1200 '                                      [9514]
1210 'Bildschirm & Farben initialisieren
                                            [D272]
1220 '                                      [B718]
1230 MODE 1:BORDER 0:INK 0,0:INK 1,26:IN
     K 2,16:INK 3,18:' bei Gruenmonitor
     auf > INK 2,22 < abaenderen !!!        [BBDC]
1240 WINDOW 1,40,24,25:WINDOW#1,14,27,6,
     20:WINDOW#2,1,40,1,23:PAPER#1,1:PEN
     #1,0:PAPER#2,0:PEN#2,1:PEN 1:PAPER
     0                                      [2FAA]
1250 '                                      [921E]
1260 'Menue                                 [2714]
1270 '                                      [9422]
1280 z=0:WINDOW#1,14,27,6,20:PRINT#1,rah
     men$;:WINDOW#1,15,26,7,19:FOR i=1 T
     O 7:LOCATE#1,1,i*2-1:PRINT#1,m$(i):
     NEXT:wahl=1                            [CFEE]
1290 LOCATE#1,1,wahl*2-1:PEN#1,3:PRINT#1
     ,m$(wahl):PEN#1,0                      [19B2]
1300 z=z+1:jo=JOY(0):IF z>2000 THEN 3650
      ELSE IF jo=0 THEN 1300                [1DCA]
1310 IF (jo AND 1)=1 AND wahl<>1 THEN FO
     R p=1 TO 100:NEXT:LOCATE#1,1,wahl*2
     -1:PRINT#1,m$(wahl):wahl=wahl-1:GOT
     O 1290                                 [1662]
1320 IF (jo AND 2)=2 AND wahl<>7 THEN FO
     R p=1 TO 100:NEXT:LOCATE#1,1,wahl*2
     -1:PRINT#1,m$(wahl):wahl=wahl+1:GOT
     O 1290                                 [A670]
1330 IF (jo AND 16)<>16 THEN 1300           [FD9C]
1340 WHILE JOY(0)<>0:WEND                   [BB0C]
1350 ON wahl GOTO 2400,1930,1990,2050,21
     00,2180,2270                           [71B2]
1360 '                                      [B522]
1370 'Unterprogramme                        [4AD4]
1380 '                                      [BB26]
1390 'SUB: Bild Nr.level loeschen (ohne
     nachruecken)                           [C958]
1400 '                                      [B918]
1410 FOR i%=FNadr(level)TO FNadr(level+1
     )-1:POKE i%,0:NEXT:FOR i=1 TO 5:top
     5$(level,i)="?.?.?":top5(level,i)=0
     :NEXT:RETURN                           [F1EE]
1420 '                                      [971C]
1430 'SUB: Bild Nr.level loeschen (mit n
     achruecken)                            [008E]
1440 '                                      [9520]
1450 IF level=PEEK(bilder)THEN RETURN EL
     SE |T,FNadr(level+1),FNadr(level),(
     PEEK(bilder)-level)*923:FOR i=level
      TO PEEK(bilder)-1:FOR j=1 TO 5:top
     5$(i,j)=top5$(i+1,j):top5(i,j)=top5
     (i+1,j):NEXT j,i:RETURN                [D84A]
1460 '                                      [9324]
1470 'SUB: Bildnr. eingeben                 [EB7A]
1480 '                                      [E128]
1490 CLS:PRINT:PRINT TAB(16)"Bildnr.:01"
     :level=1                               [196A]
1500 LOCATE 24,2:PRINT FNz$(level,2):CLS
     #2:|A,FNadr(level),40000               [B094]
1510 IF INKEY(47)<>-1 THEN level=0:RETUR
     N ELSE jo=JOY(0):IF jo=0 THEN 1510     [9C1E]
1520 IF ((jo AND 1)=1 OR (jo AND 4)=4)AN
     D level<>1 THEN level=level-1:GOTO
     1500                                   [BEA0]
1530 IF ((jo AND 2)=2 OR (jo AND 8)=8)AN
     D level<>PEEK(bilder) THEN level=le
     vel+1:GOTO 1500                        [0720]
1540 IF (jo AND 16)<>16 THEN 1510           [32A8]
1550 WHILE JOY(0)<>0:WEND:RETURN            [6646]
1560 '                                      [9526]
1570 'SUB: Bildnr.: level editieren         [0B58]
```

Listing 1. Im Basic-Hauptprogramm finden Sie sich dank der Kommentierung sicher leicht zurecht

```
1580 '                                        [DF2A]
1590 CLS:FOR i=0 TO 11:|P,i*2,192,i,0:NE
     XT:LOCATE 14,2:PRINT um$(1)" "um$(2
     )" "um$(3);:LOCATE 34,1:PRINT"BILD"
     :LOCATE 35,2:PRINT FNz$(level,2);    [CE3E]
1591 CLS#2:|A,FNadr(level),40000:wahl=0:
     xp=0:yp=176:PLOT 614,6,1:DRAWR 20,0
     :DRAWR 0,20:DRAWR -20,0:DRAWR 0,-20
     :flag=0                              [34D8]
1600 IF wahl<12 THEN LOCATE wahl+1,1:PRI
     NT CHR$(241)ELSE LOCATE 14+(wahl-12
     )*6,2:PRINT CHR$(24)um$(wahl-11)CHR
     $(24);                               [E7FA]
1610 FOR p=1 TO 100:NEXT                  [7474]
1620 jo=JOY(0):IF jo=0 THEN 1620          [73B2]
1630 IF wahl<12 THEN LOCATE wahl+1,1:PRI
     NT" "ELSE LOCATE 14+(wahl-12)*6,2:P
     RINT um$(wahl-11);                   [02D6]
1640 IF (jo AND 8)=8 AND wahl<>14 THEN w
     ahl=wahl+1:GOTO 1600                 [28B8]
1650 IF (jo AND 4)=4 AND wahl<>0 THEN wa
     hl=wahl-1:GOTO 1600                  [A844]
1660 IF (jo AND 16)<>16 THEN GOTO 1600    [AE60]
1670 WHILE JOY(0)<>0:WEND                 [7018]
1680 IF wahl<12 THEN |P,77,188,wahl ELSE
      ON wahl-11 GOTO 1780,1820,1850      [F6FC]
1690 |P,xp,yp,25:FOR p=1 TO 100:NEXT      [8A08]
1700 IF INKEY(47)<>-1 THEN GOTO 1600 ELS
     E jo=JOY(0):IF jo=0 THEN 1700        [2A24]
1710 |P,xp,yp,PEEK(FNpadr(level,xp,yp))   [449A]
1720 IF (jo AND 16)=16 THEN POKE FNpadr(
     level,xp,yp),wahl:|P,xp,yp,wahl      [6896]
1730 IF (jo AND 1)=1 AND yp<>0 THEN yp=y
     p-8                                  [E0FE]
1740 IF (jo AND 2)=2 AND yp<>176 THEN yp
     =yp+8                                [1EDC]
1750 IF (jo AND 8)=8 AND xp<>78 THEN xp=
     xp+2                                 [0C86]
1760 IF (jo AND 4)=4 AND xp<>0 THEN xp=x
     p-2                                  [07FE]
1770 GOTO 1690                            [B830]
1780 PEN 2:LOCATE 14,1:PRINT"Bild loesch
     en [J/N]":PEN 1:a$="":WHILE a$<>"j"
     AND a$<>"n":a$=LOWER$(INKEY$):WEND:
     LOCATE 14,1:PRINT SPC(20):IF a$="n"
     THEN 1600                            [C85C]
1790 PEN 2:LOCATE 14,1:PRINT"Weitermache
     n [J/N]":PEN 1:a$="":WHILE a$<>"j"A
     ND a$<>"n"AND a$<>CHR$(32):a$=LOWER
     $(INKEY$):WEND:LOCATE 14,1:PRINT SP
     C(20):IF a$=" "THEN 1600             [6CF4]
1800 IF a$="j"THEN GOSUB 1410:CLS#2:|A,F
     Nadr(level),40000:GOTO 1600          [4CFE]
1810 GOSUB 1450:POKE bilder,MAX(0,PEEK(b
     ilder)-1):level=PEEK(bilder):RETURN
                                          [EBCA]
1820 x=PEEK(FNadr(level)+920):y=PEEK(FNa
     dr(level)+921):|P,x,y,PEEK(FNpadr(l
     evel,x,y))                           [FD9E]
1830 |P,xp,yp,12:PEN 2:LOCATE 14,1:PRINT
     "Maenn. setzen [J/N]":a$="":WHILE a
     $<>"j"AND a$<>"n":a$=LOWER$(INKEY$)
     :WEND:PEN 1:LOCATE 14,1:PRINT SPC(2
     0):IF a$="n"THEN |P,xp,yp,PEEK(FNpa
     dr(level,xp,yp)):                    [8C52]
1840 POKE FNadr(level)+920,xp:POKE FNadr
     (level)+921,yp:flag=1:GOTO 1600      [C6FE]
1850 IF flag=0 THEN LOCATE 14,1:PRINT CH
     R$(24)CHR$(7)" erst Maenn.setzen "C
     HR$(24);:FOR p=1 TO 1000:NEXT:LOCAT
     E 14,1:PRINT SPC(20):GOTO 1600       [7694]
1860 LOCATE 14,1:PEN 2:PRINT"Fertig [J/N
     ]":PEN 1:a$="":WHILE a$<>"j"AND a$<
     >"n":a$=LOWER$(INKEY$):WEND:LOCATE
     14,1:PRINT SPC(20):IF a$="n"THEN GO
     TO 1600                              [8DF6]
1870 cr=0:FOR i=FNadr(level) TO FNadr(le
     vel)+919:IF PEEK(i)=2 THEN cr=cr+1   [F336]
1880 NEXT:IF cr=0 THEN LOCATE 14,1:PRINT
      CHR$(24)" min.1 Edelstein "CHR$(24
     ):FOR p=1 TO 1000:NEXT:LOCATE 14,1:
     PRINT SPC(20):GOTO 1600              [7512]
1890 POKE FNadr(level)+922,cr:RETURN      [39B0]
1900 'Interrupt fuer Zeit                 [50D8]
1910 ze=ze-1:PLOT ze*2+480,0,1:DRAWR 0,1
     4:IF ze=0 THEN xyz=REMAIN(0):RETURN
      ELSE RETURN                         [8C3E]
1920 FOR pause=1 TO 100:NEXT:RETURN       [640C]
1930 '                                    [C028]
1940 'KONSTRUIEREN                        [447C]
1950 '                                    [C22C]
1960 IF PEEK(bilder)=maxbilder THEN 1280
      ELSE level=PEEK(bilder)+1:POKE bil
     der,level:GOSUB 1410                 [942E]
1970 GOSUB 1590                           [BDC0]
1980 GOTO 1280                            [6E2C]
1990 '                                    [9634]
2000 'EDITIEREN                           [9244]
2010 '                                    [9914]
2020 IF PEEK(bilder)=0 THEN GOTO 1280 EL
     SE GOSUB 1490:IF level=0 THEN 1280   [C6F4]
2030 GOSUB 1590                           [3CA8]
2040 GOTO 1280                            [6D14]
2050 '                                    [951C]
2060 'TEST                                [6B9E]
2070 '                                    [9720]
2080 IF PEEK(bilder)=0 THEN GOTO 1280     [5702]
2090 testflag=1:GOSUB 1490:IF level=0 TH
     EN GOTO 1280 ELSE GOTO 2460          [2BD4]
2100 '                                    [BC14]
2110 'ABSPEICHERN                         [265E]
2120 '                                    [9A18]
2130 CLS:PRINT" (B)ilder oder (H)ightime
     s abspeichern?":a$="":WHILE INKEY$<
     >"":WEND:WHILE a$="":a$=LOWER$(INKE
     Y$):WEND:IF a$<>"b"AND a$<>"h"THEN
     1280                                 [60B0]
2140 IF a$="h"THEN CLS:GOSUB 2310:IF txt
     $=""THEN 1280 ELSE OPENOUT"!"+txt$+
     ".txt":FOR i=1 TO maxbilder:FOR j=1
      TO 5:WRITE#9,top5$(i,j),top5(i,j):
     NEXT j,i:CLOSEOUT:GOTO 1280          [1EA0]
2150 IF PEEK(bilder)=0 THEN GOTO 1280     [77FE]
2160 WHILE INKEY$<>"":WEND:CLS:GOSUB 231
     0:IF txt$=""THEN 1280                [B420]
2170 SAVE"!"+txt$+".pic",b,bilder,1+PEEK
     (bilder)*923:GOTO 1280               [F090]
2180 '                                    [E424]
2190 'LADEN                               [3AEE]
2200 '                                    [BE16]
2210 CLS:PRINT" (B)ilder oder (H)ightime
     s laden?":a$="":WHILE INKEY$<>"":WE
     ND:WHILE a$="":a$=LOWER$(INKEY$):WE
     ND:IF a$<>"b"AND a$<>"h"THEN 1280    [B4AE]
2220 IF a$="h"THEN CLS:GOSUB 2310:IF txt
     $=""THEN 1280 ELSE OPENIN"!"+txt$+"
     .txt":FOR i=1 TO maxbilder:FOR j=1
     TO 5:INPUT#9,top5$(i,j),top5(i,j):N
     EXT j,i:CLOSEIN:GOTO 1280            [2024]
2230 WHILE INKEY$<>"":WEND:CLS:GOSUB 231
     0:IF LEN(txt$)>8 OR txt$=""THEN 128
     0                                    [9B32]
2240 PRINT"Alte Bilder loeschen (j/n) ??
     ?":a$="":WHILE a$<>"j"AND a$<>"n"AN
     D a$<>CHR$(32):a$=LOWER$(INKEY$):WE
     ND:IF a$="j" THEN LOAD"!"+txt$+".pi
     c",bilder:GOTO 2260                  [B2B4]
2250 IF a$=CHR$(32)THEN GOTO 1280 ELSE l
     adr=bilder+1+PEEK(bilder)*923:LOAD"
     !"+txt$+".pic",ladr:banz=PEEK(ladr)
     :|T,ladr+1,ladr,banz*923:POKE bilde
     r,PEEK(bilder)+banz:GOTO 1280        [655A]
2260 FOR i=1 TO PEEK(bilder):FOR j=1 TO
     5:top5$(i,j)="?.?.?":top5(i,j)=0:NE
     XT j,i:GOTO 1280                     [0212]
2270 '                                    [9724]
2280 'ENDE                                [A95E]
2290 '                                    [E528]
2300 WHILE INKEY$<>"":WEND:MODE 2:PRINT"
     Neustart mit RUN moeglich":END       [CFA4]
2310 '                                    [C11A]
2320 'Filename eingeben                   [7A18]
2330 '                                    [C31E]
2340 PRINT"Filename:";:txt$=""            [D31E]
2350 PRINT"_"CHR$(8);:a$="":WHILE (a$<"0
     "OR a$>"Z")AND a$<>CHR$(13)AND a$<>
     CHR$(127):a$=UPPER$(INKEY$):WEND     [4352]
2360 IF a$=CHR$(13) THEN PRINT CHR$(32):
     RETURN                               [48FE]
2370 IF a$=CHR$(127) THEN IF txt$="" THE
     N PRINT CHR$(7);:GOTO 2350 ELSE txt
     $=LEFT$(txt$,LEN(txt$)-1):PRINT CHR
     $(32)CHR$(8)CHR$(8);:GOTO 2350       [EDFE]
2380 IF LEN(txt$)=8 THEN PRINT CHR$(7);:
     GOTO 2350                            [5904]
2390 PRINT a$;:txt$=txt$+a$:GOTO 2350     [F7CE]
2400 '                                    [921A]
2410 'SPIELEN                             [183C]
2420 '                                    [941E]
2430 IF PEEK(bilder)=0 THEN GOTO 1280     [7900]
2440 lev=1:GOSUB 1490:score=0:left=7:tes
     tflag=0                              [72C8]
2450 'Hier einsprung bei Test             [B54A]
2460 PEN 0:PAPER 1:CLS:PRINT" PUNKTE<3>L
     EBEN<2>BILD ENERGIE ZEIT:":IF testf
     lag=0 THEN LOCATE 1,2:PRINT"0000000
     0 "FNl$(left)                        [CFCE]
2470 'Hier einsprung bei neuem Level      [6872]
2480 fall=0:jump=0:en=100:ze=40:LOCATE 1
     9,2:PRINT FNz$(level,2):LOCATE 25,2
     :PRINT FNz$(en,3):LOCATE 31,2:PRINT
      SPC(10);:FOR i=0 TO ze*2-2 STEP 2:
     PLOT i+480,0,2:DRAWR 0,14:NEXT       [710C]
2490 CLS#2:|A,FNadr(level),40000:xp=PEEK
     (FNadr(level)+920):yp=PEEK(FNadr(le
     vel)+921):cr=PEEK(FNadr(level)+922)
     :|G,xp,yp,22:SOUND 1,0,0,0,1,0,1:|P
     ,xp,yp,27,3:WHILE SQ(1)<>4:WEND:|X,
     xp,yp,22,xp,yp,12:man=12:m=0         [2F94]
2500 zxy=FRE(""):EVERY 100 GOSUB 1910:GO
     TO 2710                              [FFBE]
2510 IF ze=0 THEN GOTO 2910 ELSE IF INKE
```

```
     Y(47)<>-1 THEN GOTO 3280 ELSE IF fa
     ll>4 THEN 3010 ELSE jo=JOY(0):IF jo
     =0 THEN 2510                              [A0D8]
2520 fall=0                                    [CDEA]
2530 IF (jo AND 16)=16 THEN 2870               [8E42]
2540 IF (jo AND 1)<>1 THEN 2580                [7EE2]
2550 IF yp=0 THEN GOTO 2510                    [CD02]
2560 vorder=PEEK(FNscreen(xp,yp-8)):IF h
     inter<>9 OR vorder=1 OR vorder=4 OR
      vorder=5 THEN 2510                       [AC00]
2570 SOUND 2,1000,2,7:man=20:m=1-m:yp=yp
     -8:|X,xp,yp+8,22,xp,yp+4,man+m:GOSU
     B 1920:|X,xp,yp+4,22,xp,yp,man+m:GO
     TO 2710                                   [FADA]
2580 IF (jo AND 2)<>2 THEN 2610                [BFE2]
2590 vorder=PEEK(FNscreen(xp,yp+8)):IF v
     order=1 OR vorder=4 OR vorder=5 THE
     N 2510                                    [63C6]
2600 SOUND 2,1000,2,7:man=20:m=1-m:yp=yp
     +8:|X,xp,yp-8,22,xp,yp-4,man+m:GOSU
     B 1920:|X,xp,yp-4,22,xp,yp,man+m:GO
     TO 2710                                   [0DD6]
2610 IF (jo AND 8)<>8 THEN 2660                [A7F8]
2620 IF xp=78 THEN GOTO 2510                   [FF7A]
2630 vorder=PEEK(FNscreen(xp+2,yp)):IF v
     order=1 OR vorder=4 OR vorder=5 THE
     N 2510                                    [EBB0]
2640 xp=xp+2:m=1-m:IF vorder=10 THEN man
     =18 ELSE man=14                           [F36C]
2650 SOUND 2,1000,2,7:|X,xp-2,yp,22,xp-1
     ,yp,man+m:GOSUB 1920:|X,xp-1,yp,22,
     xp,yp,man+m:GOTO 2710                     [84E8]
2660 IF (jo AND 4)<>4 THEN 2510                [16E6]
2670 IF xp=0 THEN GOTO 2510                    [FA06]
2680 vorder=PEEK(FNscreen(xp-2,yp)):IF v
     order=1 OR vorder=4 OR vorder=5 THE
     N 2510                                    [FDBE]
2690 xp=xp-2:m=1-m:IF vorder=10 THEN man
     =16 ELSE man=12                           [7972]
2700 SOUND 2,1000,2,7:|X,xp+2,yp,22,xp+1
     ,yp,man+m:GOSUB 1920:|X,xp+1,yp,22,
     xp,yp,man+m:GOTO 2710                     [63D4]
2710 IF yp=176 THEN GOTO 2930                  [B3E6]
2720 hinter=PEEK(FNscreen(xp,yp)):unten=
     PEEK(FNscreen(xp,yp+8))                   [96CC]
2730 IF ze=0 THEN GOTO 2910                    [13F6]
2740 IF hinter=2 THEN 2950                     [54A8]
2750 IF hinter=3 THEN 3000                     [B592]
2760 IF hinter=6 THEN 3020                     [A89E]
2770 IF hinter=7 THEN 3040                     [0BA6]
2780 IF hinter=8 THEN 3060                     [46AE]
2790 ON jump GOTO 3130,3160:jump=0             [BF08]
2800 IF unten=4 THEN ri=-1:ma=0:GOTO 308
     0                                         [AED8]
2810 IF unten=5 THEN ri=1:ma=78:GOTO 308
     0                                         [0800]
2820 IF unten=11 AND hinter<>9 AND hinte
     r<>10 THEN yp=yp+8:SOUND 1,yp+100,0
     ,15,7,3:man=20:m=1-m:|X,xp,yp-8,22,
     xp,yp-4,man+m:GOSUB 1920:|X,xp,yp-4
     ,22,xp,yp,man+m:GOSUB 1920:WHILE SQ
     (1)<>4:WEND:fall=0:GOTO 2710              [51D6]
2830 IF unten=1 OR unten=9 OR hinter=9 O
     R hinter=10 THEN 2510                     [04AC]
2840 IF man=12 THEN man=16 ELSE IF man=1
     4 THEN man=18                             [211E]
2850 yp=yp+8:SOUND 1,yp+100,0,15,6:|X,xp
     ,yp-8,22,xp,yp-4,man:GOSUB 1920:|X,
     xp,yp-4,22,xp,yp,man                      [7426]
2860 en=MAX(0,en-1):LOCATE 25,2:PRINT FN
     z$(en,3):WHILE SQ(1)<>4:WEND:fall=f
     all+1:GOTO 2710                           [BB68]
2870 IF ((jo AND 4)<>4 AND (jo AND 8)<>8
     )OR en=0 THEN GOTO 2540                   [8BBC]
2880 IF (jo AND 4)=4 THEN ri=-1 ELSE ri=
     1                                         [6034]
2890 vorder=PEEK(FNscreen(xp+ri*2,yp+8))
     :IF vorder=2 OR vorder=0 OR(ri=-1 A
     ND xp=0)OR(ri=1 AND xp=78)THEN GOTO
      2540                                     [16DC]
2900 SOUND 1,0,0,15,5,0,30:en=MAX(en-10,
     0):LOCATE 25,2:PRINT FNz$(en,3):|G,
     xp+ri*2,yp,24:FOR i=1 TO 8:|X,xp+ri
     *2,yp+i-1,24,xp+ri*2,yp+i,27:CALL &
     BD19:NEXT:|P,xp+ri*2,yp+8,0,0:|P,xp
     ,yp,man+m,3:POKE FNscreen(xp+ri*2,y
     p+8),0:WHILE SQ(1)<>4:WEND:GOTO 251
     0                                         [B9BE]
2910 'Zeit alle                                [A6DA]
2920 WHILE SQ(1)<>4 OR SQ(4)<>4:WEND:FOR
      i=100 TO 1000 STEP 5:SOUND 1,i,1,7
     :SOUND 4,i+1,1,7:NEXT:SOUND 1,1005,
     0,15,2:SOUND 4,1006,0,15,2:WHILE SQ
     (1)<>4:WEND:GOTO 3200                     [2500]
2930 'aus dem Screen gefallen                  [4C24]
2940 |P,xp,yp,22,0:FOR i=800 TO 1000 STE
     P 5:SOUND 1,i,1,7:NEXT:SOUND 1,i,0,
     15,2:WHILE SQ(1)<>4:WEND:GOTO 3200        [14C4]
2950 'Edelstein                                [A468]
2960 SOUND 1,100,4,7:SOUND 1,110,4,7:|P,
     xp,yp,0,0:|G,xp,yp,22:|P,xp,yp,man+
     m,3:cr=cr-1:POKE FNscreen(xp,yp),0:
     score=score+10*level-9:IF testflag=
     0 THEN LOCATE 1,2:PRINT FNz$(score,
     8)                                        [7248]
2970 IF cr<>0 THEN 2710                        [C5B2]
2980 xyz=REMAIN(0):RESTORE 3300:READ t,l
     :WHILE t<>0 AND JOY(0)<>16:SOUND 1,
     t,l,7:SOUND 4,t+1,l,7:READ t,l:WHIL
     E SQ(1)<>4:WEND:WEND:GOSUB 3410:IF
     testflag=1 THEN 3280                      [B876]
2990 WHILE ze<>0:GOSUB 1910:score=score+
     level:LOCATE 1,2:PRINT FNz$(score,8
     ):WEND:level=level+1:IF level>PEEK(
     bilder)THEN level=1:GOTO 2480 ELSE
     GOTO 2480                                 [6720]
3000 'Bombe                                    [F1DE]
3010 SOUND 1,0,0,15,2,0,15:|P,xp,yp,27,0
     :WHILE SQ(1)<>4:WEND:GOTO 3200            [70B6]
3020 'Zeitbonus                                [4F9E]
3030 DI:zb=MIN(80,ze+(80-ze*2)):zb=MAX(z
     e+1,zb):FOR i=ze TO zb-1:PLOT i*2+4
     80,0,2:DRAWR 0,14:SOUND 1,100-i,1,6
     :NEXT:ze=zb:EI:|P,xp,yp,0,0:|G,xp,y
     p,22:|P,xp,yp,man+m,3:POKE FNscreen
     (xp,yp),0:GOTO 2710                       [7798]
3040 'Energiebonus                             [80E8]
3050 en=MIN(999,en+50):SOUND 4,284,200,1
     ,3,1:|P,xp,yp,0,0:|G,xp,yp,22:|P,xp
     ,yp,man+m,3:POKE FNscreen(xp,yp),0:
     LOCATE 25,2:PRINT FNz$(en,3):GOTO 2
     710                                       [18C2]
3060 'Energieabzug                             [67D0]
3070 SOUND 1,0,1,7,0,0,15:en=MAX(en-10,0
     ):LOCATE 25,2:PRINT FNz$(en,3):GOTO
      2800                                     [1270]
3080 'Sprung                                   [5022]
3090 SOUND 1,100,0,15,4,2:|P,xp,yp+8,26,
     0:|X,xp,yp,22,xp,yp+2,man+1,3:GOSUB
      1920:IF ri=1 THEN man=14 ELSE man=
     12                                        [6F16]
3100 |X,xp,yp+2,22,xp,yp,man:|P,xp,yp+8,
     man/2-2,0:|P,xp,yp,man,3                  [E3D6]
3110 'Phase 1                                  [1E9C]
3120 xp=xp+ri*2:yp=yp-8:|X,xp-ri*2,yp+8,
     22,xp-ri,yp+4,man:GOSUB 1920:|X,xp-
     ri,yp+4,22,xp,yp,man:GOSUB 1920:jum
     p=1:GOTO 2730                             [94AE]
3130 'Phase 2 (jump=1)                         [23D8]
3140 vorder=PEEK(FNscreen(xp+ri*2,yp)):I
     F vorder=1 OR vorder=4 OR vorder=5
     OR xp=ma THEN jump=0:WHILE SQ(1)<>4
     :WEND:GOTO 2710                           [79F6]
3150 xp=xp+ri*2:|X,xp-ri*2,yp,22,xp-ri,y
     p,man:GOSUB 1920:|X,xp-ri,yp,22,xp,
     yp,man:GOSUB 1920:jump=2:GOTO 2710        [3814]
3160 'Phase 3 (jump=2)                         [CAE2]
3170 jump=0:vorder=PEEK(FNscreen(xp+ri*2
     ,yp+8)):IF vorder=1 OR vorder=4 OR
     vorder=5 OR xp=ma THEN WHILE SQ(1)<
     >4:WEND:GOTO 2710                         [C0C2]
3180 xp=xp+ri*2:yp=yp+8:|X,xp-ri*2,yp-8,
     22,xp-ri,yp-4,man:GOSUB 1920:|X,xp-
     ri,yp-4,22,xp,yp,man:GOSUB 1920:WHI
     LE SQ(1)<>4:WEND:GOTO 2710                [2C14]
3190 'Ein Mann weniger                         [F0D6]
3200 xyz=REMAIN(0):RESTORE 3360:READ l,t
     :WHILE t<>0:SOUND 1,t,l,7:SOUND 4,t
     +1,l,7:WHILE SQ(1)<>4:WEND:READ l,t
     :WEND                                     [3F7A]
3210 IF testflag=0 THEN left=left-1:LOCA
     TE 10,2:PRINT FNl$(left):IF left<>0
      THEN 2470                                [751C]
3220 IF testflag=1 THEN GOTO 2470              [E0E6]
3230 'Game Over                                [838A]
3240 LOCATE#2,16,15:PRINT#2,"GAME-OVER":
     WHILE INKEY$="":WEND:SOUND 1,500,0,
     15,2:SOUND 4,501,0,15,2:WHILE SQ(1)
     <>4:WEND:PEN 1:PAPER 0:GOTO 1280          [CC5A]
3280 IF testflag=1 THEN xyz=REMAIN(0):PE
     N 1:PAPER 0:SOUND 1,20,5,7:GOTO 128
     0                                         [8CD8]
3290 GOTO 3200                                 [B918]
3300 DATA 426,10,379,10,338,20,284,20,28
     4,30,253,10,284,20,338,20,426,30          [8D10]
3310 DATA 379,10,338,20,338,20,379,20,42
     6,20,379,80,426,10,379,10,338,20,28
     4,20                                      [88F0]
3320 DATA 284,30,253,10,284,20,338,20,42
     6,30,379,10,338,20,338,20,379,20,37
     9,20                                      [84DE]
3330 DATA 426,80,319,40,319,40,253,20,25
     3,40,284,20,284,20,338,20,426,20,37
     9,60                                      [A9DC]
3340 DATA 426,10,379,10,338,20,284,20,28
     4,30,253,10,284,20,338,20,426,30,37
     9,10                                      [49D0]
3350 DATA 338,20,338,20,379,20,379,20,42
     6,80,0,0                                  [B2EE]
3360 DATA 50,1016,37,1016,12,1016,50,101
```

```
     6,25,850,25,899,25,899,25,1016,25,1
     016,25,1136,100,1016,0,0                [1080]
3370 '                                       [B628]
3380 'besten 5 Liste                         [8B18]
3390 '                                       [B82C]
3400 'Zeit berechnen                         [F0E8]
3410 zeit=ze+xyz/50:posi=0:IF zeit<=top5
     (level,5) THEN 3440                     [2EEC]
3420 FOR i=5 TO 1 STEP -1:IF zeit>top5(l
     evel,i)THEN posi=i                      [48A4]
3430 NEXT:IF posi<>0 THEN FOR i=5 TO pos
     i+1 STEP -1:top5$(level,i)=top5$(le
     vel,i-1):top5(level,i)=top5(level,i
     -1):NEXT:top5(level,posi)=zeit:top5
     $(level,posi)=""                        [9EC2]
3440 GOSUB 3570 'Top 5 ausgeben              [3026]
3450 IF posi=0 THEN 3560                     [9400]
3460 WINDOW SWAP 0,2:n$="":p=1:PEN 2:LOC
     ATE 19,20:PRINT CHR$(150)CHR$(154)C
     HR$(156)TAB(19)CHR$(149)" "CHR$(149
     )TAB(19)CHR$(147)CHR$(154)CHR$(153)
     ;                                       [967A]
3470 PEN 3:LOCATE 12,9+posi*2:PRINT n$;:
     PEN 1:PRINT"?";                         [E2CE]
3480 PEN 3:LOCATE 20,21:PRINT MID$(name$
     ,p,1);:PEN 1:LOCATE 5,21:PRINT RIGH
     T$(name$,MAX(15-p,0))MID$(name$,MAX
     (1,p-14),MIN(14,p-1));:LOCATE 22,21
     :PRINT MID$(name$,p+1,15)LEFT$(name
     $,MAX(0,p-15))                          [38FC]
3490 jo=JOY(0)                               [7EF2]
3500 IF (jo AND 8)=8 THEN p=p-1:WHILE p=
     0:p=30:WEND:GOTO 3480                   [4B5E]
3510 IF (jo AND 4)=4 THEN p=p+1:WHILE p=
     31:p=1:WEND:GOTO 3480                   [A650]
3520 IF (jo AND 16)<>16 THEN 3490            [EEBA]
3530 IF p=30 THEN 3550 ELSE IF p=29 THEN
      IF n$=""THEN GOTO 3490 ELSE n$=LEF
     T$(n$,LEN(n$)-1):LOCATE 13+LEN(n$),
     posi*2+9:PEN 3:PRINT".":GOTO 3470       [F412]
3540 n$=n$+MID$(name$,p,1):IF LEN(n$)<14
      THEN 3470                              [DB1C]
3550 WINDOW SWAP 0,2:top5$(level,posi)=n
     $:posi=0:GOSUB 3570                     [5BBC]
3560 WHILE JOY(0)<>16:WEND:RETURN            [6EBA]
3570 'Top 5 ausgeben                         [A610]
3580 WINDOW SWAP 0,2:PEN 2:PAPER 0:CLS:L
     OCATE 15,2:PRINT"DIE 5 BESTEN":LOCA
     TE 16,4:PRINT"ZEITEN DER":LOCATE 17
     ,6:PRINT FNz$(level,2)".EBENE";:PEN
      1:LOCATE 15,3:PRINT STRING$(12,208
     ):LOCATE 16,5:PRINT STRING$(10,208)
     :LOCATE 17,7:PRINT STRING$(8,208)       [BA40]
3590 PEN 3:LOCATE 5,9:PRINT"PLATZ: NAME:
     <10>REST ZEIT:"                         [E964]
3600 PEN 2:LOCATE 11,9:PRINT CHR$(149):L
     OCATE 26,9:PRINT CHR$(149):LOCATE 5
     ,10:PRINT STRING$(6,154)CHR$(159)ST
     RING$(14,154)CHR$(159)STRING$(10,15
     4):FOR i=11 TO 19:LOCATE 11,i:PRINT
      CHR$(149):LOCATE 26,i:PRINT CHR$(1
     49):NEXT                                [A850]
3610 FOR i=1 TO 5                            [18C0]
3620 IF i=posi THEN PEN 3 ELSE PEN 1         [D07A]
3630 LOCATE 7,i*2+9:PRINT USING"#.";i:LO
     CATE 12,i*2+9:PRINT FNtop$(level,i)
     ;                                       [F7D6]
3640 NEXT:WINDOW SWAP 0,2:RETURN             [1CC2]
3650 level=1:GOSUB 3680                      [BC3E]
3660 jo=JOY(0):IF jo<>16 THEN 3660           [89B2]
3670 xyz=REMAIN(0):GOTO 1280                 [4A66]
3680 CLS#2:|A,FNadr(level),40000:AFTER 1
     00 GOSUB 3690:RETURN                    [4950]
3690 GOSUB 3570:level=level+1:IF level>P
     EEK(bilder)THEN level=1                 [2AC0]
3700 AFTER 100 GOSUB 3680:RETURN             [C270]
```

Listing 1. Basic-Hauptprogamm (Schluß)

```
100 '*********************************** [31D4]
101 '* SRBIN.DAT - DATA-Lader von 'CPC'  * [25CA]
102 '*********************************** [A3D8]
103 '                                      [DEB6]
104 DATA A000,01,0D,A0,21,09,A0,C3,D1,164F [B736]
105 DATA A008,BC,FC,A6,0D,A0,1E,A0,C3,71EB [B616]
106 DATA A010,EE,A0,C3,08,A1,C3,12,A1,4161 [6462]
107 DATA A018,C3,91,A1,C3,A6,A1,D0,C7,5B03 [41A2]
108 DATA A020,D8,D4,C1,D3,00,32,AC,A0,4D20 [358A]
109 DATA A028,CD,45,A1,78,E5,2A,AD,A0,621A [ECD2]
110 DATA A030,01,10,00,FE,00,28,04,09,0BC1 [54EC]
111 DATA A038,3D,18,F8,E5,D1,E1,06,08,0CD8 [A8AA]
112 DATA A040,CD,50,A0,23,13,CD,50,A0,671C [9266]
113 DATA A048,2B,13,CD,26,BC,10,F1,C9,0E0B [32BE]
114 DATA A050,3A,AC,A0,FE,00,20,03,1A,2D7C [509A]
115 DATA A058,77,C9,FE,01,20,04,1A,AE,179A [68A2]
116 DATA A060,77,C9,FE,02,20,06,1A,4E,1742 [7B5A]
117 DATA A068,77,79,12,C9,FE,03,20,07,2CAB [7094]
118 DATA A070,1A,4E,CD,80,A0,77,C9,EB,0A85 [C6E4]
119 DATA A078,1A,4E,EB,CD,80,A0,77,C9,0997 [21EC]
120 DATA A080,C5,06,04,F5,E6,88,FE,00,683C [4A92]
121 DATA A088,28,10,CB,F9,CB,7F,20,02,0116 [0370]
122 DATA A090,CB,B9,CB,D9,CB,5F,20,02,5856 [5BE6]
123 DATA A098,CB,99,CB,01,F1,CB,07,10,5E0A [A2D4]
124 DATA A0A0,E2,79,CB,07,CB,07,CB,07,7185 [0FBE]
125 DATA A0A8,CB,07,C1,C9,03,00,A4,CD,716D [BDCA]
126 DATA A0B0,45,A1,E5,78,2A,AD,A0,01,1345 [F97A]
127 DATA A0B8,10,00,FE,00,28,04,09,3D,16BF [B368]
128 DATA A0C0,18,F8,E5,D1,E1,06,08,7E,24CE [82D2]
129 DATA A0C8,12,23,13,7E,12,13,2B,CD,0407 [8346]
130 DATA A0D0,26,BC,10,F3,C9,D5,C5,F5,3553 [D7BC]
131 DATA A0D8,AF,CD,25,A0,F1,C1,E1,E5,6FCB [D354]
132 DATA A0E0,C5,F5,AF,CD,AF,A0,F1,C1,402B [943C]
133 DATA A0E8,E1,41,CD,25,A0,C9,FE,04,7CEC [5024]
134 DATA A0F0,20,09,DD,7E,00,DD,23,DD,0DEF [6C10]
135 DATA A0F8,23,18,06,FE,03,C0,3A,AC,1B60 [61A8]
136 DATA A100,A0,CD,3B,A1,CD,25,A0,C9,6945 [62C8]
137 DATA A108,FE,03,C0,CD,3B,A1,CD,AF,6979 [872E]
138 DATA A110,A0,C9,FE,07,20,09,DD,7E,7D10 [8ABE]
139 DATA A118,00,DD,23,DD,23,18,06,FE,3F7A [63C6]
140 DATA A120,06,C0,3A,AC,A0,DD,4E,00,3868 [6B96]
141 DATA A128,DD,5E,02,DD,56,04,DD,46,77CC [00F2]
142 DATA A130,06,DD,6E,08,DD,66,0A,CD,3EA9 [5202]
143 DATA A138,D5,A0,C9,DD,46,00,DD,6E,5594 [28DA]
144 DATA A140,02,DD,66,04,C9,C5,4C,26,3F22 [807E]
145 DATA A148,00,11,08,00,CD,7F,A1,D5,0343 [FC42]
146 DATA A150,11,50,00,CD,6A,A1,D1,E5,10C3 [D06A]
147 DATA A158,21,00,08,CD,6A,A1,D1,19,1D3F [CA92]
148 DATA A160,16,00,59,19,11,00,C0,19,00A1 [15D6]
149 DATA A168,C1,C9,7B,B2,28,0B,C5,E5,5683 [32CA]
150 DATA A170,C1,1B,7A,B3,28,07,09,18,6366 [C942]
151 DATA A178,F8,21,00,00,C9,C1,C9,C5,705B [B982]
152 DATA A180,01,00,00,ED,52,38,03,03,0C25 [B0E8]
153 DATA A188,18,F9,19,E5,D1,C5,E1,C1,3BAF [8200]
154 DATA A190,C9,DD,4E,00,DD,46,01,DD,5D2F [0C08]
155 DATA A198,5E,02,DD,56,03,DD,6E,04,32F4 [E4B8]
156 DATA A1A0,DD,66,05,ED,B0,C9,DD,6E,7E00 [0128]
157 DATA A1A8,02,DD,66,03,DD,5E,00,DD,3DFD [CE24]
158 DATA A1B0,56,01,E5,DD,E1,21,00,C0,3D7C [00A6]
159 DATA A1B8,01,98,03,C5,E5,D5,DD,7E,2F08 [F8EA]
160 DATA A1C0,00,12,FE,00,28,1D,E5,2A,1B94 [1D78]
161 DATA A1C8,AD,A0,11,10,00,19,3D,20,7D9E [6A80]
162 DATA A1D0,FC,E5,D1,E1,06,08,1A,77,5323 [C69C]
163 DATA A1D8,23,13,1A,77,13,2B,CD,26,10F8 [987A]
164 DATA A1E0,BC,10,F3,D1,13,DD,23,E1,4A3B [45DA]
165 DATA A1E8,23,23,C1,0B,78,B1,20,CB,005F [558A]
166 DATA A1F0,C9,9A,9C,95,20,20,20,20,5930 [B148]
167 DATA A1F8,20,20,20,20,20,20,20,20,1FE0 [81E0]
168 DATA *ENDE*                            [4FD2]
169 adr=&A000:zeile=14:MEMORY adr-1        [06D0]
170 READ d$:IF d$="*ENDE*"THEN 181          [948E]
171 pr=0                                   [5010]
172 FOR i=1 TO 8                           [3166]
173 READ a$:a=VAL("&"+a$)                  [E244]
174 POKE adr,a:adr=adr+1                   [B920]
175 pr=pr*2:IF pr>65535 THEN pr=pr-65535   [08A0]
176 pr=UNT(pr)XOR a:IF pr<0 THEN pr=pr+6553
    6                                      [A5B8]
177 NEXT i                                 [210E]
178 READ pr$:pr2=VAL("&"+pr$):IF pr2<0 THEN
     pr2=pr2+65536                         [2B9A]
179 IF pr<>pr2 THEN PRINT"Pruefsummenfehler
     in Zeile";zeile:STOP                  [E722]
180 zeile=zeile+1:GOTO 170                 [035E]
181 SAVE"SR.BIN",B,&A000,&200              [A19C]
```

Listing 2. Ganze 512 Byte bilden die Maschinencode-Routinen

```
100 '*********************************** [31D4]
101 '* SRSPR.DAT - DATA-Lader von 'CPC'  * [7902]
102 '*********************************** [A3D8]
103 '                                      [DEB6]
104 DATA A400,00,00,00,00,00,00,00,00,0000 [A320]
105 DATA A408,00,00,00,00,00,00,00,00,0000 [4C32]
106 DATA A410,07,07,07,07,07,07,00,00,02F4 [2EB2]
107 DATA A418,0B,0B,0B,0B,0B,0B,00,00,06E4 [1A4E]
108 DATA A420,00,00,00,00,00,00,00,00,0000 [EF2C]
109 DATA A428,70,E0,53,AC,53,AC,70,E0,0088 [CB7E]
110 DATA A430,10,00,00,80,10,00,37,CE,0020 [DB9C]
111 DATA A438,37,CE,17,8E,17,8E,17,8E,2020 [6B7C]
112 DATA A440,00,30,00,C0,30,00,C1,08,000A [33B2]
113 DATA A448,03,0C,07,0E,0F,0F,00,00,02C4 [0426]
114 DATA A450,C0,00,30,00,00,C0,01,38,653A [5AD4]
115 DATA A458,03,0C,07,0E,0F,0F,00,00,02C4 [402C]
116 DATA A460,10,80,31,40,51,20,91,10,291A [F0B4]
117 DATA A468,91,FE,40,20,20,40,10,80,7DA0 [F230]
118 DATA A470,00,00,30,C0,20,40,20,00,0A40 [409E]
119 DATA A478,30,80,20,00,20,40,30,C0,3CA0 [B8EA]
```

Listing 3. Sprite-Routinen zur Bewegung unseres Helden »Bity«

```
120 DATA A480,00,00,00,00,00,00,00,00,0000  [8B2C]
121 DATA A488,00,00,00,00,01,01,05,05,0003  [BC5C]
122 DATA A490,44,22,FF,FF,44,22,44,22,3892  [7856]
123 DATA A498,44,22,FF,FF,44,22,44,22,3892  [0D68]
124 DATA A4A0,F0,F0,00,00,00,00,00,00,4400  [2DAE]
125 DATA A4A8,00,00,00,00,00,00,00,00,0000  [7858]
126 DATA A4B0,00,88,11,00,00,88,11,00,2222  [0FA4]
127 DATA A4B8,00,88,11,00,00,88,11,00,2222  [54B6]
128 DATA A4C0,10,80,10,80,00,80,30,A0,20C0  [ADD8]
129 DATA A4C8,50,C0,10,80,20,40,60,C0,1200  [6DEA]
130 DATA A4D0,10,80,10,80,00,80,50,C0,2060  [F2BA]
131 DATA A4D8,30,A0,10,80,A0,40,40,80,3E00  [E304]
132 DATA A4E0,10,80,10,80,10,00,50,C0,22E0  [A8D4]
133 DATA A4E8,30,A0,10,80,20,40,30,60,3A00  [AADE]
134 DATA A4F0,10,80,10,80,10,00,30,A0,2240  [81B0]
135 DATA A4F8,50,C0,10,80,20,50,10,20,1240  [2CC8]
136 DATA A500,50,A0,50,A0,40,A0,30,C0,0020  [F7DE]
137 DATA A508,10,80,10,80,20,40,60,C0,2200  [09A8]
138 DATA A510,90,A0,90,A0,40,A0,30,C0,7820  [5112]
139 DATA A518,10,80,10,80,20,40,40,80,2200  [9594]
140 DATA A520,50,A0,50,A0,50,20,30,C0,02A0  [ADDE]
141 DATA A528,10,80,10,80,20,40,30,60,2200  [3A82]
142 DATA A530,50,90,50,90,50,20,30,C0,0DA0  [B1E8]
143 DATA A538,10,80,10,80,20,40,10,20,2200  [DC7C]
144 DATA A540,10,C0,10,C0,40,90,30,E0,36C0  [7C04]
145 DATA A548,10,C0,10,C0,10,40,30,60,3780  [ACD4]
146 DATA A550,10,C0,10,C0,00,80,30,E0,3480  [76E6]
147 DATA A558,10,C0,10,C0,20,20,60,30,3770  [C5D6]
148 DATA A560,0B,0B,0B,0B,00,00,00,00,0690  [E1EC]
149 DATA A568,00,00,00,00,00,00,00,00,0000  [E950]
150 DATA A570,F0,C0,60,60,60,60,70,C0,40A0  [3712]
151 DATA A578,60,C0,60,60,E0,60,00,00,0C80  [CEFA]
152 DATA A580,44,22,FF,FF,44,22,44,22,3892  [8F5C]
153 DATA A588,44,22,FF,FF,44,22,44,22,3892  [506E]
154 DATA A590,10,80,10,80,10,80,E0,70,2130  [45B8]
155 DATA A598,E0,70,10,80,10,80,10,80,6420  [D3D6]
156 DATA A5A0,00,00,00,00,F0,F0,01,08,044A  [E6EE]
157 DATA A5A8,03,0C,07,0E,0F,0F,00,00,02C4  [C352]
158 DATA A5B0,10,00,41,40,00,00,95,9A,0590  [38D8]
159 DATA A5B8,11,88,00,04,42,20,10,20,2850  [6ECA]
160 DATA *ENDE*                              [BFC2]
161 adr=&A400:zeile=104:MEMORY adr-1        [FC28]
162 READ d$:IF d$="*ENDE*"THEN 173          [AD92]
163 pr=0                                    [5E12]
164 FOR i=1 TO 8                            [1368]
165 READ a$:a=VAL("&"+a$)                   [F946]
166 POKE adr,a:adr=adr+1                    [8D22]
167 pr=pr*2:IF pr>65535 THEN pr=pr-65535    [18A2]
168 pr=UNT(pr)XOR a:IF pr<0 THEN pr=pr+6553
    6                                       [55BA]
169 NEXT i                                  [3310]
170 READ pr$:pr2=VAL("&"+pr$):IF pr2<0 THEN
     pr2=pr2+65536                          [AB8A]
171 IF pr<>pr2 THEN PRINT"Pruefsummenfehler
     in Zeile";zeile:STOP                   [3F12]
172 zeile=zeile+1:GOTO 162                  [2A62]
173 SAVE"SR.SPR".B.&A400.&1C0               [8302]
```

Listing 3. Sprite-Routinen (Schluß)

```
1000 DATA 61A0,33,2C,30,30,2C,30,30,09,1649
1001 DATA 61A8,00,00,00,00,02,02,02,00,001C
1002 DATA 61B0,0A,00,02,02,02,00,00,00,0570
1003 DATA 61B8,00,00,00,00,00,0A,0A,0A,0036
1004 DATA 61C0,0A,0A,0A,0A,02,00,00,00,0670
1005 DATA 61C8,00,00,00,0A,0A,00,00,00,00F0
1006 DATA 61D0,00,00,00,01,01,01,01,01,001F
1007 DATA 61D8,00,01,01,01,01,01,00,09,0075
1008 DATA 61E0,01,01,01,01,01,0B,09,01,00C7
1009 DATA 61E8,02,00,00,01,01,00,01,01,011B
1010 DATA 61F0,09,01,01,00,03,09,00,00,04DC
1011 DATA 61F8,00,00,00,01,01,00,00,00,0018
1012 DATA 6200,00,00,00,01,01,00,05,09,001B
1013 DATA 6208,01,00,00,01,01,0B,09,01,00A7
1014 DATA 6210,01,01,02,01,01,00,01,01,009B
1015 DATA 6218,09,00,00,00,00,09,00,0B,04AF
1016 DATA 6220,00,00,00,01,01,09,01,09,0037
1017 DATA 6228,00,00,00,01,01,00,00,09,0011
1018 DATA 6230,01,00,00,01,01,0B,09,01,00A7
1019 DATA 6238,01,01,01,01,01,00,01,01,00FB
1020 DATA 6240,01,09,05,00,01,01,01,0B,0265
1021 DATA 6248,00,00,00,00,00,09,01,09,002F
1022 DATA 6250,0A,0A,0A,00,00,00,0A,09,06DD
1023 DATA 6258,01,02,01,01,01,0B,01,01,0017
1024 DATA 6260,00,01,01,01,01,00,01,01,007B
1025 DATA 6268,01,09,02,00,00,00,00,0B,028B
1026 DATA 6270,00,00,00,00,09,01,01,01,004F
1027 DATA 6278,00,00,00,01,01,01,00,09,0015
1028 DATA 6280,01,01,01,01,01,0B,01,01,00D7
1029 DATA 6288,00,00,01,01,01,00,01,01,003B
1030 DATA 6290,01,01,01,09,00,00,00,0B,007B
1031 DATA 6298,00,00,09,01,01,01,01,01,013F
1032 DATA 62A0,00,00,00,01,01,01,00,09,0015
1033 DATA 62A8,01,01,01,01,01,0B,01,01,00D7
1034 DATA 62B0,00,00,00,01,01,00,01,01,001B
1035 DATA 62B8,01,01,01,09,00,00,00,0B,007B
1036 DATA 62C0,00,00,09,00,00,00,00,00,0120
1037 DATA 62C8,00,00,00,00,00,00,00,00,0000
1038 DATA 62D0,00,00,00,00,00,00,00,00,0000
1039 DATA 62D8,00,00,00,00,00,00,00,00,0000
1040 DATA 62E0,00,00,00,09,00,00,00,0B,009B
1041 DATA 62E8,00,00,09,00,00,00,00,00,0120
1042 DATA 62F0,00,00,00,00,00,00,00,00,0000
1043 DATA 62F8,00,00,00,02,00,00,00,00,0020
1044 DATA 6300,00,00,00,00,00,00,00,03,0003
1045 DATA 6308,00,00,00,09,00,00,00,0B,009B
1046 DATA 6310,00,01,01,09,01,00,00,01,00F9
1047 DATA 6318,00,00,09,05,00,01,00,00,0174
1048 DATA 6320,00,01,00,01,09,00,00,01,0019
1049 DATA 6328,00,01,09,01,01,05,09,01,017F
1050 DATA 6330,01,01,01,09,00,00,00,0B,007B
1051 DATA 6338,00,01,00,09,00,01,00,01,00D5
1052 DATA 6340,00,00,09,01,00,01,05,00,013E
1053 DATA 6348,00,01,00,01,01,09,00,01,007D
1054 DATA 6350,00,01,09,00,00,00,09,09,017B
1055 DATA 6358,09,00,00,01,00,00,00,0B,049B
1056 DATA 6360,00,01,00,09,00,01,00,01,00D5
1057 DATA 6368,00,00,09,01,00,01,00,01,0135
1058 DATA 6370,00,01,00,01,00,01,09,01,0047
1059 DATA 6378,00,01,09,02,00,00,00,01,0141
1060 DATA 6380,09,02,00,01,0B,00,00,0B,0443
1061 DATA 6388,00,01,01,01,09,00,00,01,0039
1062 DATA 6390,00,00,09,01,00,01,00,01,0135
1063 DATA 6398,00,01,00,01,00,01,09,01,0047
1064 DATA 63A0,00,01,01,01,01,09,00,01,005D
1065 DATA 63A8,01,01,09,00,0B,00,00,0B,01B3
1066 DATA 63B0,00,01,00,01,09,00,00,01,0019
1067 DATA 63B8,00,00,09,01,00,01,00,01,0135
1068 DATA 63C0,03,01,00,01,00,01,09,00,01C6
1069 DATA 63C8,0A,0A,00,00,00,09,00,01,07A5
1070 DATA 63D0,00,01,09,00,0B,00,00,0B,0133
1071 DATA 63D8,00,01,00,00,01,09,0A,0A,0072
1072 DATA 63E0,00,02,09,01,00,01,00,00,01B4
1073 DATA 63E8,01,01,00,01,00,00,01,01,00D3
1074 DATA 63F0,00,01,02,02,02,09,00,01,0015
1075 DATA 63F8,00,00,01,09,0A,00,00,0B,00EB
1076 DATA 6400,00,01,00,00,00,01,00,00,0044
1077 DATA 6408,01,01,01,00,00,01,00,00,00E4
1078 DATA 6410,00,01,00,01,00,00,00,01,0051
1079 DATA 6418,00,01,01,01.01,01,00,01,007D
1080 DATA 6420,00,00,00,01,00,00,00,0B,001B
1081 DATA 6428,00,00,00,00,00,00,00,00,0000
1082 DATA 6430,00,00,00,00,00,00,00,00,0000
1083 DATA 6438,00,00,00,00,00,00,00,00,0000
1084 DATA 6440,00,00,00,00,00,00,00,00,0000
1085 DATA 6448,00,00,00,00,00,00,00,0B,000B
1086 DATA 6450,00,00,00,00,00,00,00,00,0000
1087 DATA 6458,00,00,00,00,00,00,00,00,0000
1088 DATA 6460,00,00,00,00,00,00,00,00,0000
1089 DATA 6468,00,00,00,00,00,00,00,00,0000
1090 DATA 6470,00,00,00,00,00,00,00,0B,000B
1091 DATA 6478,00,00,00,00,00,00,00,00,0000
1092 DATA 6480,00,00,00,00,00,00,00,00,0000
1093 DATA 6488,00,00,00,00,00,00,00,00,0000
1094 DATA 6490,00,00,00,00,00,00,00,00,0000
1095 DATA 6498,00,00,00,00,00,00,00,06,0006
1096 DATA 64A0,00,00,00,00,00,00,00,00,0000
1097 DATA 64A8,00,00,00,00,00,00,00,00,0000
1098 DATA 64B0,00,00,00,00,00,00,00,00,0000
1099 DATA 64B8,00,00,00,00,00,00,00,00,0000
1100 DATA 64C0,00,00,00,00,00,00,00,02,0002
1101 DATA 64C8,00,00,00,00,00,00,00,00,0000
1102 DATA 64D0,00,00,00,00,00,00,00,00,0000
1103 DATA 64D8,00,00,00,00,00,00,00,00,0000
1104 DATA 64E0,00,00,00,00,00,00,00,00,0000
1105 DATA 64E8,00,00,00,00,00,00,00,00,0000
1106 DATA 64F0,03,00,03,00,03,00,03,00,01FE
1107 DATA 64F8,03,00,03,00,03,00,03,00,01FE
1108 DATA 6500,03,00,03,00,03,00,03,00,01FE
1109 DATA 6508,03,00,03,00,03,00,03,00,01FE
1110 DATA 6510,03,00,03,00,03,00,03,00,01FE
1111 DATA 6518,00,03,00,03,00,03,00,03,00FF
1112 DATA 6520,00,03,00,03,00,03,00,03,00FF
1113 DATA 6528,00,03,00,03,00,03,00,03,00FF
1114 DATA 6530,00,03,00,03,00,03,00,03,00FF
1115 DATA 6538,00,03,00,03,00,03,00,03,00FF
1116 DATA 6540,06,00,13,00,00,00,00,00,0160
1117 DATA 6548,00,00,00,00,00,00,00,01,0001
1118 DATA 6550,01,01,00,00,00,01,01,00,00C6
1119 DATA 6558,00,00,00,00,00,00,02,00,0004
1120 DATA 6560,00,00,00,0A,0A,00,00,00,00F0
1121 DATA 6568,02,00,00,00,00,00,00,00,0100
1122 DATA 6570,00,00,00,01,01,01,09,09,0007
1123 DATA 6578,09,09,09,09,09,09,09,09,0707
```

Listing 4. Wer sich die Arbeit sparen möchte, dieses Listing abzutippen, gibt die Bilder über den Editor ein

```
1124 DATA 6580,01,01,01,01,01,01,01,01,00FF
1125 DATA 6588,01,01,01,00,00,01,01,01,00E7
1126 DATA 6590,01,01,01,00,00,00,00,00,00E0
1127 DATA 6598,00,00,04,01,00,00,01,01,0093
1128 DATA 65A0,00,00,00,09,01,00,00,01,0099
1129 DATA 65A8,00,00,00,00,00,00,00,00,0000
1130 DATA 65B0,00,00,00,0A,0A,00,00,00,00F0
1131 DATA 65B8,00,00,00,00,00,00,00,02,0002
1132 DATA 65C0,00,00,00,00,00,00,00,00,0000
1133 DATA 65C8,01,01,00,09,00,00,01,00,0052
1134 DATA 65D0,00,00,00,00,00,00,00,00,0000
1135 DATA 65D8,00,00,0A,0A,0A,0A,00,00,0198
1136 DATA 65E0,00,00,00,00,00,00,02,01,0005
1137 DATA 65E8,02,00,00,00,00,00,00,00,0100
1138 DATA 65F0,00,00,01,09,00,00,00,01,00B1
1139 DATA 65F8,00,00,00,00,00,00,00,00,0000
1140 DATA 6600,0A,0A,0A,0A,0A,0A,0A,0A,0606
1141 DATA 6608,00,00,00,00,00,02,01,00,000A
1142 DATA 6610,01,02,00,00,00,00,00,00,0000
1143 DATA 6618,00,00,00,01,09,00,00,00,0058
1144 DATA 6620,00,00,00,00,00,00,01,02,0000
1145 DATA 6628,00,00,0A,0A,0A,0A,00,00,0198
1146 DATA 6630,00,00,00,00,02,01,00,00,0014
1147 DATA 6638,00,01,02,00,00,00,00,00,0000
1148 DATA 6640,00,00,00,00,01,00,00,04,000C
1149 DATA 6648,09,09,01,00,00,01,00,01,06E5
1150 DATA 6650,02,00,00,0A,0A,00,00,00,01F0
1151 DATA 6658,00,00,00,02,01,0B,00,00,0004
1152 DATA 6660,00,0B,01,02,00,00,00,00,02C0
1153 DATA 6668,00,00,00,00,00,00,00,01,0001
1154 DATA 6670,09,09,01,00,01,00,00,00,06E8
1155 DATA 6678,01,02,00,0A,0A,00,00,00,00F0
1156 DATA 6680,00,00,00,09,00,0B,00,00,00BC
1157 DATA 6688,00,0B,00,01,0B,00,00,00,0288
1158 DATA 6690,00,00,00,00,00,00,00,01,0001
1159 DATA 6698,09,09,01,01,00,00,00,00,06F0
1160 DATA 66A0,00,01,02,0A,0A,00,00,00,00F0
1161 DATA 66A8,00,00,00,01,00,0B,00,00,003C
1162 DATA 66B0,00,0B,00,01,0B,00,00,00,0288
1163 DATA 66B8,00,00,00,00,00,00,00,01,0001
1164 DATA 66C0,09,09,00,00,00,00,00,00,06C0
1165 DATA 66C8,00,00,01,02,0A,00,00,00,0050
1166 DATA 66D0,00,00,00,00,01,00,00,00,0008
1167 DATA 66D8,00,02,01,00,0B,00,00,00,00F8
1168 DATA 66E0,00,00,00,00,00,00,00,01,0001
1169 DATA 66E8,09,01,01,01,01,01,01,01,04FF
1170 DATA 66F0,09,01,00,01,02,00,00,00,04C0
1171 DATA 66F8,00,00,00,00,00,01,02,00,0000
1172 DATA 6700,02,01,00,00,0B,00,00,00,0118
1173 DATA 6708,00,00,00,00,00,00,00,01,0001
1174 DATA 6710,01,00,00,00,00,00,00,00,0080
1175 DATA 6718,09,00,00,00,01,02,00,00,0480
1176 DATA 6720,00,00,00,00,00,00,01,09,000B
1177 DATA 6728,05,00,00,00,0B,00,00,00,02D8
1178 DATA 6730,00,00,00,00,00,00,00,01,0001
1179 DATA 6738,00,00,00,00,00,00,00,00,0000
1180 DATA 6740,09,00,00,00,00,01,02,00,0480
1181 DATA 6748,00,00,00,00,00,00,00,09,0009
1182 DATA 6750,00,00,00,00,0B,00,00,00,0058
1183 DATA 6758,00,00,00,00,00,00,01,01,0003
1184 DATA 6760,09,01,01,01,01,01,01,01,04FF
1185 DATA 6768,01,01,01,01,01,01,01,02,00FC
1186 DATA 6770,00,00,00,00,00,00,00,09,0009
1187 DATA 6778,00,00,00,00,0B,00,00,00,0058
1188 DATA 6780,00,00,00,00,00,01,01,00,0006
1189 DATA 6788,09,00,00,00,00,00,00,00,0480
1190 DATA 6790,00,00,00,00,00,00,01,01,0003
1191 DATA 6798,02,00,00,00,00,00,00,09,0109
1192 DATA 67A0,00,01,00,00,0B,00,00,00,0018
1193 DATA 67A8,00,00,00,00,01,01,01,00,000E
1194 DATA 67B0,09,00,00,00,00,00,00,00,0480
1195 DATA 67B8,00,00,00,00,00,00,01,01,0003
1196 DATA 67C0,01,0B,00,00,00,00,00,09,0249
1197 DATA 67C8,01,00,00,00,0B,00,00,00,00D8
1198 DATA 67D0,00,00,00,00,00,01,01,01,0007
1199 DATA 67D8,01,01,01,01,01,01,01,09,00F7
1200 DATA 67E0,01,01,01,01,01,01,01,00,00FE
1201 DATA 67E8,00,0B,00,00,00,00,00,09,02C9
1202 DATA 67F0,00,00,00,00,0B,00,00,00,0058
1203 DATA 67F8,00,00,00,00,00,01,00,00,0004
1204 DATA 6800,00,00,00,00,00,00,00,09,0009
1205 DATA 6808,00,00,00,00,00,00,01,00,0002
1206 DATA 6810,00,0B,00,00,00,00,00,09,02C9
1207 DATA 6818,00,00,00,00,0B,00,00,00,0058
1208 DATA 6820,00,00,00,00,00,00,00,00,0000
1209 DATA 6828,00,00,00,00,00,00,00,09,0009
1210 DATA 6830,00,00,00,00,00,00,01,00,0002
1211 DATA 6838,00,0B,00,00,00,00,00,09,02C9
1212 DATA 6840,00,00,00,00,0B,00,00,00,0058
1213 DATA 6848,00,00,00,00,00,09,01,01,0027
1214 DATA 6850,01,01,01,01,01,01,01,01,00FF
1215 DATA 6858,01,01,01,09,00,00,01,00,0072
1216 DATA 6860,00,0B,00,00,00,00,00,09,02C9
1217 DATA 6868,00,00,00,00,00,00,00,00,0000
1218 DATA 6870,00,00,00,00,09,09,01,00,006E
1219 DATA 6878,00,05,09,00,00,00,00,00,0060
1220 DATA 6880,00,00,00,09,00,00,01,00,0092
1221 DATA 6888,00,0B,00,00,00,00,00,09,02C9
1222 DATA 6890,00,00,00,00,00,00,00,00,0000
1223 DATA 6898,00,00,00,09,09,06,01,02,00C0
1224 DATA 68A0,02,09,09,00,00,00,00,02,0262
1225 DATA 68A8,02,02,02,09,02,02,00,00,0148
1226 DATA 68B0,00,0B,00,01,01,01,01,01,02DF
1227 DATA 68B8,01,01,01,01,01,01,01,01,00FF
1228 DATA 68C0,01,01,01,01,01,01,01,01,00FF
1229 DATA 68C8,01,01,01,00,00,00,00,01,00E1
1230 DATA 68D0,01,01,01,01,01,01,01,01,00FF
1231 DATA 68D8,01,01,01,1A,A8,20,00,00,0480
1232 DATA 68E0,00,00,00,00,00,00,00,00,0000
1233 DATA 68E8,00,00,00,0A,0A,0A,0A,0A,00C6
1234 DATA 68F0,0A,0A,0A,0A,0A,00,00,00,0630
1235 DATA 68F8,00,00,00,00,00,00,00,00,0000
1236 DATA 6900,0B,0A,0A,0A,0A,00,0A,0A,06AE
1237 DATA 6908,00,00,00,00,0A,00,00,0A,005A
1238 DATA 6910,0A,0A,0A,0A,0A,0A,0A,0A,0606
1239 DATA 6918,00,00,00,00,0A,0A,0A,0A,0066
1240 DATA 6920,0A,0A,00,00,00,00,00,0A,078A
1241 DATA 6928,0B,0A,0A,00,00,00,00,0A,064A
1242 DATA 6930,0A,00,00,00,0A,0A,0A,0A,0566
1243 DATA 6938,00,0A,0A,0A,00,0A,00,00,0348
1244 DATA 6940,00,00,00,00,00,00,00,0A,000A
1245 DATA 6948,00,0A,00,00,00,00,0A,00,0294
1246 DATA 6950,0B,00,0C,00,00,00,00,00,0580
1247 DATA 6958,00,0A,00,0A,0A,0A,00,0A,0252
1248 DATA 6960,00,0A,00,0A,0A,00,00,00,0270
1249 DATA 6968,00,01,01,00,00,00,00,0A,006A
1250 DATA 6970,00,0A,0A,0A,0A,0A,00,0A,0312
1251 DATA 6978,0B,0A,0A,00,00,00,00,00,0640
1252 DATA 6980,00,0A,00,0A,0A,0A,0A,00,024C
1253 DATA 6988,00,0A,0A,00,00,00,00,00,03C0
1254 DATA 6990,01,01,01,01,00,00,00,00,00F0
1255 DATA 6998,0A,0A,00,00,00,00,0A,00,0794
1256 DATA 69A0,0A,0A,0A,0B,00,00,00,00,0670
1257 DATA 69A8,0A,0A,00,0A,0A,0A,00,00,0758
1258 DATA 69B0,00,00,00,00,00,00,00,01,0001
1259 DATA 69B8,01,00,00,01,01,00,00,00,0098
1260 DATA 69C0,00,00,0A,00,00,00,0A,0A,015E
1261 DATA 69C8,0A,0A,00,0B,00,00,00,0A,073A
1262 DATA 69D0,00,00,0A,0A,00,00,0A,0A,01FE
1263 DATA 69D8,00,00,00,00,00,00,01,01,0003
1264 DATA 69E0,00,00,00,02,01,01,00,00,002C
1265 DATA 69E8,00,00,0A,0A,0A,0A,00,0A,0192
1266 DATA 69F0,0A,0A,0A,0B,00,00,0A,00,0664
1267 DATA 69F8,00,00,0A,0A,0A,0A,0A,0A,0186
1268 DATA 6A00,00,00,00,00,00,01,01,01,0007
1269 DATA 6A08,01,09,01,01,01,01,01,00,02FE
1270 DATA 6A10,00,00,0A,0A,0A,0A,00,00,0198
1271 DATA 6A18,00,0A,0A,0B,00,00,0A,00,0364
1272 DATA 6A20,00,0A,00,00,00,00,00,00,0280
1273 DATA 6A28,00,00,00,00,01,01,00,00,000C
1274 DATA 6A30,00,09,00,00,00,00,01,01,0243
1275 DATA 6A38,00,00,00,00,00,00,0A,00,0014
1276 DATA 6A40,0A,0A,0A,0A,00,00,0A,00,0674
1277 DATA 6A48,00,0A,00,00,00,00,00,00,0280
1278 DATA 6A50,00,00,00,01,01,00,00,00,0018
1279 DATA 6A58,00,09,00,00,00,00,00,01,0241
1280 DATA 6A60,01,00,00,00,00,00,00,0A,008A
1281 DATA 6A68,0B,0A,0A,0A,00,00,0A,0A,06FE
1282 DATA 6A70,0A,0A,0A,0A,0A,00,00,00,0630
1283 DATA 6A78,00,00,01,01,01,09,01,01,001F
1284 DATA 6A80,01,01,01,01,01,01,01,01,00FF
1285 DATA 6A88,01,01,00,00,00,00,00,00,00C0
1286 DATA 6A90,0B,0A,0A,00,00,00,00,00,0640
1287 DATA 6A98,00,0A,0A,0A,0A,00,00,00,0330
1288 DATA 6AA0,00,01,01,00,00,09,00,00,0044
1289 DATA 6AA8,00,00,00,00,00,00,00,00,0000
1290 DATA 6AB0,00,01,01,00,00,00,00,00,0060
1291 DATA 6AB8,0B,0A,0A,0A,0A,0B,00,00,069C
1292 DATA 6AC0,0A,00,0A,00,00,00,00,00,0440
1293 DATA 6AC8,01,01,00,00,00,09,00,00,00E4
1294 DATA 6AD0,00,00,00,00,00,00,00,00,0000
1295 DATA 6AD8,00,00,01,01,00,00,00,00,0030
1296 DATA 6AE0,0B,00,00,00,0A,0B,00,00,05FC
1297 DATA 6AE8,0A,0A,00,00,00,00,00,01,0781
1298 DATA 6AF0,01,01,01,01,01,01,01,01,00FF
1299 DATA 6AF8,01,01,01,01,01,01,01,01,00FF
1300 DATA 6B00,01,09,01,01,01,00,00,00,02F8
1301 DATA 6B08,0B,00,00,0A,00,0B,0A,0A,0512
1302 DATA 6B10,00,0A,00,00,00,00,01,01,0283
1303 DATA 6B18,00,00,00,00,00,00,01,01,0003
1304 DATA 6B20,01,01,01,01,01,00,00,00,00F8
1305 DATA 6B28,00,09,00,00,01,01,00,00,024C
1306 DATA 6B30,0B,0A,0A,00,00,0B,0A,00,0678
1307 DATA 6B38,00,0A,00,00,00,01,01,01,0287
```

```
1308 DATA 6B40,01,01,01,01,01,01,01,01,00FF
1309 DATA 6B48,09,01,01,01,01,00,00,00,04F8
1310 DATA 6B50,00,09,00,00,00,01,01,00,0246
1311 DATA 6B58,0B,00,0A,0A,00,0B,0A,0A,0452
1312 DATA 6B60,0A,0A,00,00,01,01,00,00,078C
1313 DATA 6B68,00,00,00,00,02,01,01,01,0017
1314 DATA 6B70,09,01,01,01,01,09,01,01,04DF
1315 DATA 6B78,01,01,01,01,01,01,01,01,00FF
1316 DATA 6B80,0B,00,00,0A,00,00,0A,0A,053E
1317 DATA 6B88,00,00,00,01,01,01,01,01,001F
1318 DATA 6B90,09,01,01,01,01,01,01,02,04FC
1319 DATA 6B98,02,02,01,01,01,09,00,01,019D
1320 DATA 6BA0,01,09,00,0A,0A,0A,0A,0A,0206
1321 DATA 6BA8,0A,0A,0A,0A,00,00,0A,00,0674
1322 DATA 6BB0,00,0B,01,01,00,00,00,00,02F0
1323 DATA 6BB8,09,00,00,00,00,01,01,02,0484
1324 DATA 6BC0,02,02,01,01,01,09,00,01,019D
1325 DATA 6BC8,01,09,00,00,02,02,02,02,02DE
1326 DATA 6BD0,01,01,00,00,00,00,0A,00,00D4
1327 DATA 6BD8,00,01,01,01,01,09,01,01,005F
1328 DATA 6BE0,01,01,01,01,01,01,01,02,00FC
1329 DATA 6BE8,03,02,01,01,01,09,02,01,0119
1330 DATA 6BF0,01,09,01,01,01,01,01,01,02FF
1331 DATA 6BF8,01,01,01,00,00,00,00,00,00E0
1332 DATA 6C00,01,01,01,01,01,09,01,01,00DF
1333 DATA 6C08,01,01,01,01,01,01,01,09,00F7
1334 DATA 6C10,09,09,09,01,01,01,01,01,07FF
1335 DATA 6C18,01,09,00,00,00,00,00,00,02C0
1336 DATA 6C20,00,00,01,01,00,00,00,01,0031
1337 DATA 6C28,01,00,00,00,00,09,00,00,00A4
1338 DATA 6C30,00,00,00,00,00,00,00,00,0000
1339 DATA 6C38,00,00,00,00,00,00,00,00,0000
1340 DATA 6C40,00,09,00,00,00,02,02,02,024E
1341 DATA 6C48,02,02,03,01,01,00,01,01,01FB
1342 DATA 6C50,01,01,01,01,01,01,01,01,00FF
1343 DATA 6C58,01,01,01,01,01,01,01,01,00FF
1344 DATA 6C60,01,01,01,01,01,01,01,01,00FF
1345 DATA 6C68,01,09,01,01,01,01,01,01,02FF
1346 DATA 6C70,01,01,01,01,01,01,4C,00,0064
1347 DATA 6C78,14,00,00,00,00,00,00,00,0A00
1348 DATA 6C80,00,00,00,00,00,00,00,00,0000
1349 DATA 6C88,00,00,00,00,00,00,00,00,0000
1350 DATA 6C90,00,00,00,00,00,00,00,00,0000
1351 DATA 6C98,00,00,00,00,00,00,00,00,0000
1352 DATA 6CA0,00,01,01,01,01,01,01,01,007F
1353 DATA 6CA8,01,01,01,01,01,01,01,01,00FF
1354 DATA 6CB0,01,09,01,01,01,01,01,01,02FF
1355 DATA 6CB8,09,01,01,01,01,01,01,01,04FF
1356 DATA 6CC0,01,01,01,09,01,01,01,01,007F
1357 DATA 6CC8,01,01,00,00,00,01,00,00,00C4
1358 DATA 6CD0,00,00,00,00,00,01,00,00,0004
1359 DATA 6CD8,00,09,00,00,00,00,01,00,0242
1360 DATA 6CE0,09,00,01,00,00,00,00,00,04A0
1361 DATA 6CE8,00,00,00,09,00,00,00,00,0090
1362 DATA 6CF0,01,01,02,00,00,01,00,00,0084
1363 DATA 6CF8,00,00,02,00,00,01,00,00,0044
1364 DATA 6D00,00,09,00,00,02,00,01,00,0252
1365 DATA 6D08,09,02,01,00,00,00,02,00,0424
1366 DATA 6D10,00,00,02,09,00,00,00,00,00D0
1367 DATA 6D18,01,01,01,09,01,01,01,01,007F
1368 DATA 6D20,09,01,01,01,01,01,01,09,04F7
1369 DATA 6D28,01,01,01,09,01,01,01,01,007F
1370 DATA 6D30,01,01,01,01,09,01,01,01,00BF
1371 DATA 6D38,09,01,01,01,01,01,09,01,04EF
1372 DATA 6D40,01,01,00,09,00,00,00,00,0050
1373 DATA 6D48,09,00,00,01,00,00,00,09,0499
1374 DATA 6D50,00,01,00,09,00,00,00,00,00D0
1375 DATA 6D58,01,00,00,00,09,00,01,00,00CA
1376 DATA 6D60,09,00,00,01,00,00,09,00,0482
1377 DATA 6D68,01,01,00,09,00,00,00,00,0050
1378 DATA 6D70,09,00,02,01,00,02,00,09,04D1
1379 DATA 6D78,00,01,02,09,00,00,00,02,0092
1380 DATA 6D80,01,00,00,02,09,00,01,00,00EA
1381 DATA 6D88,09,00,02,01,02,00,09,00,04D2
1382 DATA 6D90,01,01,01,01,01,09,01,01,00DF
1383 DATA 6D98,01,01,01,01,01,01,01,01,00FF
1384 DATA 6DA0,01,01,01,01,01,09,01,01,00DF
1385 DATA 6DA8,01,01,09,01,01,01,01,01,01FF
1386 DATA 6DB0,01,09,01,01,01,09,01,01,02DF
1387 DATA 6DB8,01,01,00,00,00,09,00,01,00E5
1388 DATA 6DC0,00,00,00,00,01,00,00,00,0008
1389 DATA 6DC8,00,00,00,01,00,09,00,00,0034
1390 DATA 6DD0,01,00,09,00,00,00,00,01,01A1
1391 DATA 6DD8,00,09,00,01,00,09,00,00,0274
1392 DATA 6DE0,01,01,00,00,00,09,02,01,00E1
1393 DATA 6DE8,00,00,00,02,01,00,00,00,0028
1394 DATA 6DF0,00,02,00,01,00,09,00,02,00B6
1395 DATA 6DF8,01,00,09,00,00,02,00,01,01A9
1396 DATA 6E00,00,09,02,01,00,09,00,02,0236
1397 DATA 6E08,01,01,01,09,01,01,01,01,007F
1398 DATA 6E10,01,09,01,01,01,01,09,01,02EF
1399 DATA 6E18,01,01,01,01,01,01,01,01,00FF
1400 DATA 6E20,01,01,01,01,09,01,01,01,00BF
1401 DATA 6E28,01,01,01,01,01,01,09,01,00EF
1402 DATA 6E30,01,01,00,09,00,00,00,01,0051
1403 DATA 6E38,00,09,00,00,00,00,09,00,0252
1404 DATA 6E40,00,00,01,00,00,00,00,00,0020
1405 DATA 6E48,00,00,00,00,09,00,00,00,0048
1406 DATA 6E50,00,01,00,00,00,00,09,00,0052
1407 DATA 6E58,01,01,02,09,00,00,00,01,0011
1408 DATA 6E60,00,09,00,02,00,00,09,00,0272
1409 DATA 6E68,00,02,01,00,00,00,00,00,00A0
1410 DATA 6E70,02,00,00,00,09,00,02,00,014C
1411 DATA 6E78,00,01,00,00,02,00,09,00,0042
1412 DATA 6E80,01,01,01,01,01,09,01,01,00DF
1413 DATA 6E88,01,01,01,01,01,01,01,09,00F7
1414 DATA 6E90,01,01,01,01,01,01,09,01,00EF
1415 DATA 6E98,01,01,09,01,01,01,01,01,01FF
1416 DATA 6EA0,01,01,01,01,01,01,01,01,00FF
1417 DATA 6EA8,01,01,00,00,00,09,00,00,00E4
1418 DATA 6EB0,01,00,00,00,00,00,00,09,0089
1419 DATA 6EB8,00,00,00,00,00,00,09,00,0012
1420 DATA 6EC0,01,00,09,00,00,01,00,00,01A4
1421 DATA 6EC8,00,00,00,01,00,00,00,00,0010
1422 DATA 6ED0,01,01,00,00,00,09,00,02,00E6
1423 DATA 6ED8,01,00,02,00,00,00,00,09,00C9
1424 DATA 6EE0,00,02,00,00,00,00,09,00,0092
1425 DATA 6EE8,01,00,09,00,02,01,00,02,01B6
1426 DATA 6EF0,00,00,00,01,00,00,00,02,0012
1427 DATA 6EF8,01,01,01,09,01,01,01,01,007F
1428 DATA 6F00,01,01,01,01,09,01,01,01,00BF
1429 DATA 6F08,01,01,01,09,01,01,01,01,007F
1430 DATA 6F10,01,01,01,09,01,01,01,01,007F
1431 DATA 6F18,01,09,01,01,01,09,01,01,02DF
1432 DATA 6F20,01,01,00,09,00,00,00,00,0050
1433 DATA 6F28,00,00,00,00,09,00,00,01,0049
1434 DATA 6F30,00,00,00,09,00,01,00,00,0094
1435 DATA 6F38,00,00,00,09,00,00,01,00,0092
1436 DATA 6F40,00,09,00,00,00,09,00,00,0264
1437 DATA 6F48,01,01,00,09,00,08,08,02,0062
1438 DATA 6F50,08,00,08,08,09,00,02,01,05CD
1439 DATA 6F58,00,00,00,09,02,01,00,02,0086
1440 DATA 6F60,00,00,00,09,00,02,01,00,009A
1441 DATA 6F68,00,09,00,02,00,09,00,02,0246
1442 DATA 6F70,01,01,01,01,01,01,01,01,00FF
1443 DATA 6F78,01,09,01,01,01,09,01,01,02DF
1444 DATA 6F80,01,09,01,01,01,01,01,01,02FF
1445 DATA 6F88,01,09,01,01,01,01,01,01,02FF
1446 DATA 6F90,09,01,01,01,01,01,09,01,04EF
1447 DATA 6F98,01,01,00,00,00,00,00,00,00C0
1448 DATA 6FA0,00,09,00,01,00,09,00,01,0275
1449 DATA 6FA8,00,09,00,00,00,00,00,01,0241
1450 DATA 6FB0,00,09,00,00,00,00,00,00,0240
1451 DATA 6FB8,09,00,01,00,00,00,09,00,04B2
1452 DATA 6FC0,01,01,08,08,08,08,06,08,0124
1453 DATA 6FC8,08,09,00,01,02,09,00,01,0665
1454 DATA 6FD0,00,09,00,00,02,00,00,01,0251
1455 DATA 6FD8,00,09,00,00,00,02,00,00,0248
1456 DATA 6FE0,09,00,01,00,02,00,09,00,04A2
1457 DATA 6FE8,01,01,01,01,01,01,01,01,00FF
1458 DATA 6FF0,01,01,01,01,01,01,01,01,00FF
1459 DATA 6FF8,01,01,01,01,01,01,01,01,00FF
1460 DATA 7000,01,01,01,01,01,01,01,01,00FF
1461 DATA 7008,01,01,01,01,01,01,01,01,00FF
1462 DATA 7010,01,02,A8,2B,09,0A,0A,0A,17CE
1463 DATA 7018,0A,0A,0A,0A,0A,0A,0A,0A,0606
1464 DATA 7020,0A,0A,0A,0A,0A,0A,0A,0A,0606
1465 DATA 7028,0A,0A,0A,0A,0A,0A,0A,0A,0606
1466 DATA 7030,0A,0A,0A,0A,0A,0A,0A,0A,0606
1467 DATA 7038,0A,0A,0A,09,09,00,00,00,0618
1468 DATA 7040,00,00,00,00,00,00,00,00,0000
1469 DATA 7048,00,00,0B,00,00,00,00,00,0160
1470 DATA 7050,00,00,00,00,00,0B,00,00,002C
1471 DATA 7058,00,00,00,00,00,00,00,00,0000
1472 DATA 7060,00,00,00,09,09,02,00,00,00D0
1473 DATA 7068,00,02,00,00,00,02,00,00,0088
1474 DATA 7070,00,02,0B,0A,0A,0A,0A,0A,0126
1475 DATA 7078,0A,0A,0A,0A,0A,0B,02,00,0618
1476 DATA 7080,00,00,02,00,00,00,02,00,0044
1477 DATA 7088,00,00,02,09,01,01,01,05,00DB
1478 DATA 7090,01,01,01,05,01,01,01,05,00BB
1479 DATA 7098,01,01,01,00,00,00,00,00,00E0
1480 DATA 70A0,00,00,00,00,00,01,01,01,0007
1481 DATA 70A8,04,01,01,01,04,01,01,01,0257
1482 DATA 70B0,04,01,01,09,01,00,01,01,02FB
1483 DATA 70B8,01,00,01,01,01,00,01,01,00BB
1484 DATA 70C0,01,00,01,00,00,00,00,00,00A0
1485 DATA 70C8,00,00,00,00,00,01,06,01,0009
1486 DATA 70D0,01,01,00,01,01,01,00,01,00DD.
1487 DATA 70D8,01,01,00,09,01,00,00,00,0058
1488 DATA 70E0,00,00,00,00,00,00,00,00,0000
1489 DATA 70E8,00,00,01,00,00,00,00,00,0020
1490 DATA 70F0,00,00,00,00,00,01,00,00,0004
```

```
1491 DATA 70F8,00,00,00,00,00,00,00,00,0000
1492 DATA 7100,00,00,00,09,01,00,00,00,0098
1493 DATA 7108,00,00,00,00,00,00,00,00,0000
1494 DATA 7110,00,00,00,0A,0A,0A,0A,0A,00C6
1495 DATA 7118,0A,0A,0A,0A,0A,00,00,00,0630
1496 DATA 7120,00,00,00,00,08,08,00,00,0060
1497 DATA 7128,00,00,00,09,01,09,01,01,00BF
1498 DATA 7130,01,01,01,01,01,01,01,01,00FF
1499 DATA 7138,01,01,01,00,00,00,00,00,00E0
1500 DATA 7140,00,00,00,00,00,01,01,01,0007
1501 DATA 7148,01,01,01,01,01,01,01,01,00FF
1502 DATA 7150,01,01,01,01,01,09,00,00,00DC
1503 DATA 7158,00,00,00,00,00,00,00,00,0000
1504 DATA 7160,00,00,00,00,00,00,00,00,0000
1505 DATA 7168,00,00,00,00,00,00,00,00,0000
1506 DATA 7170,00,00,00,00,00,00,00,00,0000
1507 DATA 7178,00,00,00,01,01,01,01,09,0017
1508 DATA 7180,00,00,00,00,00,00,00,00,0000
1509 DATA 7188,01,01,01,0A,0A,0A,0A,0A,0026
1510 DATA 7190,0A,0A,0A,0A,0A,01,01,01,0637
1511 DATA 7198,09,00,00,00,00,00,00,00,0480
1512 DATA 71A0,00,01,01,01,00,00,01,09,007B
1513 DATA 71A8,00,00,00,00,00,00,00,00,0000
1514 DATA 71B0,00,00,00,00,00,00,07,00,000E
1515 DATA 71B8,07,00,00,00,00,00,00,00,0380
1516 DATA 71C0,09,00,00,00,00,00,00,00,0480
1517 DATA 71C8,00,01,00,00,00,00,01,01,0043
1518 DATA 71D0,01,01,01,01,01,01,01,01,00FF
1519 DATA 71D8,01,09,01,05,01,01,05,00,02B6
1520 DATA 71E0,04,01,01,04,01,01,09,01,023F
1521 DATA 71E8,01,01,01,01,01,01,01,01,00FF
1522 DATA 71F0,09,01,00,00,00,00,01,0A,04C8
1523 DATA 71F8,0A,0A,0A,0A,0A,0A,0A,09,0605
1524 DATA 7200,01,09,01,01,01,00,01,01,02FB
1525 DATA 7208,01,00,01,01,01,00,09,01,00AB
1526 DATA 7210,0A,0A,0A,0A,0A,0A,0A,0A,0606
1527 DATA 7218,09,01,00,00,00,00,01,00,04C2
1528 DATA 7220,00,00,01,01,01,00,00,09,0031
1529 DATA 7228,01,09,00,00,00,00,00,00,02C0
1530 DATA 7230,00,00,00,00,00,00,09,01,0013
1531 DATA 7238,00,00,00,01,01,01,00,00,001C
1532 DATA 7240,09,01,00,00,00,00,01,00,04C2
1533 DATA 7248,00,01,02,02,02,01,00,09,003D
1534 DATA 7250,01,09,00,00,00,01,01,01,02C7
1535 DATA 7258,01,01,01,00,00,00,09,01,00F3
1536 DATA 7260,00,00,01,02,02,02,01,00,001A
1537 DATA 7268,09,01,00,00,00,00,01,00,04C2
1538 DATA 7270,00,01,02,02,02,01,00,09,003D
1539 DATA 7278,01,09,00,00,01,09,09,09,02F7
1540 DATA 7280,09,09,09,01,00,00,09,01,07E3
1541 DATA 7288,00,00,01,02,02,02,01,00,001A
1542 DATA 7290,09,01,00,00,00,00,01,00,04C2
1543 DATA 7298,00,01,02,02,02,01,00,09,003D
1544 DATA 72A0,01,09,00,01,09,09,00,00,02BC
1545 DATA 72A8,00,00,09,09,01,00,09,01,01AB
1546 DATA 72B0,00,00,01,02,02,02,01,00,001A
1547 DATA 72B8,09,01,00,00,00,00,01,00,04C2
1548 DATA 72C0,00,01,01,01,01,01,00,09,0075
1549 DATA 72C8,01,09,00,01,09,00,00,00,0298
1550 DATA 72D0,00,00,00,09,01,00,09,01,008B
1551 DATA 72D8,00,00,01,01,01,01,01,00,003E
1552 DATA 72E0,09,01,00,00,00,00,01,00,04C2
1553 DATA 72E8,00,0B,0B,0B,0B,0B,00,09,036D
1554 DATA 72F0,01,09,00,01,09,00,00,02,029A
1555 DATA 72F8,02,00,00,09,01,00,09,01,018B
1556 DATA 7300,00,00,0B,0B,0B,0B,0B,00,01B2
1557 DATA 7308,09,01,00,00,00,00,01,00,04C2
1558 DATA 7310,00,00,00,00,00,00,00,09,0009
1559 DATA 7318,01,09,00,01,09,00.00,01,0299
1560 DATA 7320,01,00,00,09,01,00,09,01,000B
1561 DATA 7328,00,00,00,00,00,00,00,00,0000
1562 DATA 7330,09,01,00,00,00,00,01,00,04C2
1563 DATA 7338,00,00,00,00,00,00,00,09,0009
1564 DATA 7340,01,09,00,01,09,00,00,00,0298
1565 DATA 7348,00,00,00,09,01,00,09,01,008B
1566 DATA 7350,00,00,00,00,00,00,00,00,0000
1567 DATA 7358,09,01,00,00,00,00,01,00,04C2
1568 DATA 7360,00,00,00,00,00,00,00,09,0009
1569 DATA 7368,01,09,00,01,09,00,00,00,0298
1570 DATA 7370,00,00,00,09,01,00,09,01,008B
1571 DATA 7378,00,00,00,00,00,00,00,00,0000
1572 DATA 7380,09,01,00,00,01,01,01,01,04CF
1573 DATA 7388,01,01,01,01,01,01,01,01,00FF
1574 DATA 7390,01,01,01,01,01,01,01,01,00FF
1575 DATA 7398,01,01,01,01,01,01,01,01,00FF
1576 DATA 73A0,01,01,01,01,01,01,01,01,00FF
1577 DATA 73A8,01,01,01,01,28,A8,1C,01,0329
1578 DATA 73B0,01,01,01,01,01,01,01,01,00FF
1579 DATA 73B8,01,01,01,01,01,01,01,01,00FF
1580 DATA 73C0,01,01,01,01,01,01,01,01,00FF
1581 DATA 73C8,01,01,01,01,01,01,01,01,00FF
1582 DATA 73D0.01.01,01,01,01,01,01,01,00FF
1583 DATA 73D8,00,00,00,00,00,00,00,00,0000
1584 DATA 73E0,00,00,00,00,00,00,00,00,0000
1585 DATA 73E8,00,00,00,00,00,00,00,02,0002
1586 DATA 73F0,02,00,00,00,00,00,00,00,0100
1587 DATA 73F8,00,00,00,00,00,00,01,01,0003
1588 DATA 7400,00,00,00,00,00,00,00,00,0000
1589 DATA 7408,00,00,00,00,00,00,00,00,0000
1590 DATA 7410,00,00,09,09,09,01,01,01,01FF
1591 DATA 7418,01,01,01,09,00,00,00,00,0070
1592 DATA 7420,00,00,00,00,00,00,01,01,0003
1593 DATA 7428,00,00,00,04,04,00,04,04,006C
1594 DATA 7430,00,04,04,00,04,04,00,04,01B4
1595 DATA 7438,04,04,04,04,09,00,00,00,0388
1596 DATA 7440,00,00,00,09,00,00,00,00,0090
1597 DATA 7448,00,00,00,00,00,00,01,01,0003
1598 DATA 7450,00,00,00,00,00,00,00,00,0000
1599 DATA 7458,00,00,00,00,00,00,00,00,0000
1600 DATA 7460,00,00,00,00,09,00,00,00,0048
1601 DATA 7468,00,00,00,09,00,00,00,00,0090
1602 DATA 7470,00,00,00,00,00,00,01,01,0003
1603 DATA 7478,00,00,00,00,00,00,00,00,0000
1604 DATA 7480,00,00,00,00,00,00,00,00,0000
1605 DATA 7488,00,00,01,01,01,01,01,00,003E
1606 DATA 7490,00,00,00,09,00,00,00,00,0090
1607 DATA 7498,00,00,00,00,00,00,01,01,0003
1608 DATA 74A0,00,00,02,00,00,00,00,00,0040
1609 DATA 74A8,02,03,03,00,00,00,00,00,01A0
1610 DATA 74B0,00,04,00,00,00,00,00,00,0100
1611 DATA 74B8,00,00,00,09,00,00,00,00,0090
1612 DATA 74C0,00,00,00,00,00,00,01,01,0003
1613 DATA 74C8,0B,01,01,01,05,00,00,01,05D9
1614 DATA 74D0,01,01,01,01,01,01,01,00,00FE
1615 DATA 74D8,00,00,00,00,00,00,00,03,0003
1616 DATA 74E0,03,00,00,09,00,02,02,00,011C
1617 DATA 74E8,00,00,00,00,00,00,01,01,0003
1618 DATA 74F0,0B,00,00,00,00,00,00,01,0581
1619 DATA 74F8,03,02,02,02,03,01,00,00,017C
1620 DATA 7500,00,00,00,00,01,01,01,01,000F
1621 DATA 7508,01,01,01,01,01,01,01,01,00FF
1622 DATA 7510,09,00,00,00,00,00,01,01,0483
1623 DATA 7518,0B,00,00,00,00,00,00,01,0581
1624 DATA 7520,01,01,01,01,01,01,00,00,00FC
1625 DATA 7528,00,00,00,00,00,00,00,00,0000
1626 DATA 7530,00,00,00,00,00,00,00,00,0000
1627 DATA 7538,09,00,00,00,00,00,01,01,0483
1628 DATA 7540,0B,00,00,00,00,00,00,00,0580
1629 DATA 7548,00,00,00,00,00,00,00,0A,000A
1630 DATA 7550,0A,0A,0A,0B,00,00,00,00,0670
1631 DATA 7558,00,00,02,00,00,00,00,00,0040
1632 DATA 7560,09,00,00,00,00,00,01,01,0483
1633 DATA 7568,0B,00,00,00,00,00,09,01,0593
1634 DATA 7570,01,01,01,01,01,01,01,00,00FE
1635 DATA 7578,00,00,00,0B,00,00,00,00,00B0
1636 DATA 7580,00,01,01,01,09,01,01,09,0037
1637 DATA 7588,01,00,00,00,00,00,01,01,0083
1638 DATA 7590,0B,00,00,00,00,00,09,00,0592
1639 DATA 7598,00,00,00,00,00,00,00,00,0000
1640 DATA 75A0,00,00,00,0B,00,00,00,00,00B0
1641 DATA 75A8,00,00,00,00,09,00,00,09,0041
1642 DATA 75B0,00,00,00,00,00,00,01,01,0003
1643 DATA 75B8,0B,00,00,00,09,01,01,01,05CF
1644 DATA 75C0,01,09,00,00,00,00,03,03,02C5
1645 DATA 75C8,03,00,00,0B,00,00,00,00,0130
1646 DATA 75D0,00,00,00,00,09,00,00,09,0041
1647 DATA 75D8,00,02,00,00,00,00,01,01,0083
1648 DATA 75E0,0B,00,00,00,09,00,00,00,05C8
1649 DATA 75E8,00,09,00,00,00,01,01,01,0247
1650 DATA 75F0,01,01,01,09,01,01,00,00,007C
1651 DATA 75F8,00,00,00,00,09,00,01,01,004B
1652 DATA 7600,01,01,01,00,00,00,01,01,00E3
1653 DATA 7608,0B,00,00,00,09,00,00,00,05C8
1654 DATA 7610,00,09,00,00,00,00,00,00,0240
1655 DATA 7618,00,00,00,09,00,00,00,00,0090
1656 DATA 7620,00,00,00,00,09,00,00,00,0048
1657 DATA 7628,00,00,00,00,00,00,01,01,0003
1658 DATA 7630,0B,00,00,00,09,00,00,02,05CA
1659 DATA 7638,00,09,00,00,00,00,00,00,0240
1660 DATA 7640,00,00,00,09,00,03,03,00,009A
1661 DATA 7648,00,00,00,02,09,00,00,00,0068
1662 DATA 7650,00,00,00,00,00,00,01,01,0003
1663 DATA 7658,0B,00,00,00,09,00,01,01,05CB
1664 DATA 7660,01,01,01,00,00,00,00,00,00E0
1665 DATA 7668,00,00,00,01,01,01,01,01,001F
1666 DATA 7670,01,09,01,01,01,00,00,00,02F8
1667 DATA 7678,00,00,00,00,00,00,01,01,0003
1668 DATA 7680,0B,00,02,00,09,00,00,00,0588
1669 DATA 7688,00,00,00,00,00,00,00,00,0000
1670 DATA 7690,00,00,00,00,00,00,00,00,0000
1671 DATA 7698,00,09,00,00,00,09,09,09,027F
1672 DATA 76A0,09,09,09,09,00,00,01,01,0773
1673 DATA 76A8,0B,00,01,01,01,09,00,00,059C
1674 DATA 76B0,00,00,00,00,00,00,00,00,0000
```

```
1675 DATA 76B8,00,00,00,00,0B,00,00,00,0058
1676 DATA 76C0,00,09,00,00,00,09,01,01,0267
1677 DATA 76C8,01,01,01,09,00,00,01,01,0073
1678 DATA 76D0,0B,00,00,00,00,09,00,00,05A4
1679 DATA 76D8,00,00,00,00,00,00,00,00,0000
1680 DATA 76E0,00,00,00,00,00,00,00,00,0000
1681 DATA 76E8,00,09,00,00,00,09,01,01,0267
1682 DATA 76F0,01,01,01,09,00,00,01,01,0073
1683 DATA 76F8,0B,00,00,00,00,09,00,00,05A4
1684 DATA 7700,00,00,00,00,00,02,02,02,000E
1685 DATA 7708,02,02,02,02,02,02,02,02,01FE
1686 DATA 7710,00,09,00,00,00,09,01,01,0267
1687 DATA 7718,02,01,01,09,00,00,01,01,01F3
1688 DATA 7720,01,01,01,01,01,01,01,01,00FF
1689 DATA 7728,01,01,01,01,01,01,01,01,00FF
1690 DATA 7730,01,01,01,03,01,01,01,01,00DF
1691 DATA 7738,01,01,01,01,01,01,01,01,00FF
1692 DATA 7740,01,01,01,01,01,01,01,0A,00F4
1693 DATA 7748,A8,1A,01,08,08,08,08,08,5258
1694 DATA 7750,08,08,08,0B,00,00,00,00,07B0
1695 DATA 7758,00,00,00,00,00,0A,0A,0A,0036
1696 DATA 7760,0A,0A,0A,00,00,00,00,00,06C0
1697 DATA 7768,00,00,00,00,00,00,00,00,0000
1698 DATA 7770,00,00,01,01,01,01,01,01,003F
1699 DATA 7778,01,01,01,0B,00,00,00,00,0050
1700 DATA 7780,00,00,00,00,0A,0A,01,09,0073
1701 DATA 7788,01,0A,0A,0A,00,00,00,00,03E0
1702 DATA 7790,00,00,00,00,00,00,00,00,0000
1703 DATA 7798,00,00,00,00,00,00,00,00,0000
1704 DATA 77A0,00,00,01,0B,00,00,00,00,0090
1705 DATA 77A8,00,00,00,0A,0A,0A,01,09,00D3
1706 DATA 77B0,01,0A,0A,0A,0A,00,00,00,03B0
1707 DATA 77B8,00,00,00,00,00,00,00,00,0000
1708 DATA 77C0,00,00,01,01,01,01,01,01,003F
1709 DATA 77C8,01,01,01,0B,00,00,00,00,0050
1710 DATA 77D0,00,00,0A,0A,0A,0A,01,09,0193
1711 DATA 77D8,01,0A,0A,0A,0A,0A,00,00,0398
1712 DATA 77E0,00,00,00,00,00,00,00,00,0000
1713 DATA 77E8,00,00,00,00,00,00,00,00,0000
1714 DATA 77F0,01,01,00,0B,00,00,00,00,0070
1715 DATA 77F8,00,0A,0A,0A,0A,0A,01,09,0313
1716 DATA 7800,01,0A,0A,0A,0A,0A,0A,00,038C
1717 DATA 7808,00,00,00,00,00,00,00,00,0000
1718 DATA 7810,00,00,01,01,01,01,01,01,003F
1719 DATA 7818,01,00,00,0B,00,00,00,00,0030
1720 DATA 7820,0A,0A,0A,0A,0A,0A,01,09,0613
1721 DATA 7828,01,0A,0A,0A,0A,0A,0A,0A,0386
1722 DATA 7830,00,00,00,00,00,00,00,00,0000
1723 DATA 7838,00,00,00,00,00,00,01,01,0003
1724 DATA 7840,00,00,00,0B,00,00,00,0A,00BA
1725 DATA 7848,0A,0A,0A,0A,0A,0A,01,09,0613
1726 DATA 7850,01,0A,0A,0A,0A,0A,0A,0A,0386
1727 DATA 7858,0A,00,00,00,00,00,00,00,0500
1728 DATA 7860,00,00,01,01,01,01,01,00,003E
1729 DATA 7868,00,00,00,00,00,00,0A,0A,001E
1730 DATA 7870,0A,0A,02,0A,0A,0A,01,09,0713
1731 DATA 7878,01,0A,02,02,02,02,02,0A,0276
1732 DATA 7880,0A,0A,00,00,00,00,00,00,0780
1733 DATA 7888,00,00,07,00,01,01,00,00,00EC
1734 DATA 7890,00,00,00,00,00,0A,0A,0A,0036
1735 DATA 7898,0A,02,02,0A,0A,0A,01,09,0513
1736 DATA 78A0,01,0A,0A,0A,0A,02,0A,0A,03A6
1737 DATA 78A8,0A,0A,0A,00,00,00,00,00,06C0
1738 DATA 78B0,00,00,01,01,01,00,00,00,0038
1739 DATA 78B8,00,00,00,00,0A,0A,0A,0A,0066
1740 DATA 78C0,02,0A,02,0A,0A,0A,01,09,0313
1741 DATA 78C8,01,0A,0A,0A,02,0A,0A,0A,03C6
1742 DATA 78D0,0A,0A,0A,0A,00,00,00,00,0660
1743 DATA 78D8,00,00,01,01,01,00,00,00,0038
1744 DATA 78E0,00,00,00,0A,0A,0A,0A,02,00CE
1745 DATA 78E8,0A,0A,02,0A,0A,0A,01,09,0713
1746 DATA 78F0,01,0A,0A,0A,02,0A,0A,0A,03A6
1747 DATA 78F8,0A,0A,0A,0A,0A,00,00,00,0630
1748 DATA 7900,00,00,01,01,01,00,00,00,0038
1749 DATA 7908,00,00,0A,0A,0A,0A,0A,0A,0186
1750 DATA 7910,0A,0A,02,0A,0A,0A,01,09,0713
1751 DATA 7918,01,0A,0A,0A,0A,0A,02,0A,0396
1752 DATA 7920,0A,0A,0A,0A,0A,0A,00,00,0618
1753 DATA 7928,00,00,01,01,01,00,00,00,0038
1754 DATA 7930,0A,0A,0A,0A,0A,0A,0A,0A,0606
1755 DATA 7938,0A,0A,02,0A,0A,0A,01,09,0713
1756 DATA 7940,01,0A,02,0A,0A,0A,02,0A,0296
1757 DATA 7948,0A,0A,0A,0A,0A,0A,0A,00,060C
1758 DATA 7950,00,00,01,01,01,00,0A,0A,0026
1759 DATA 7958,0A,0A,0A,0A,0A,0A,0A,0A,0606
1760 DATA 7960,0A,02,02,02,0A,0A,01,09,0593
1761 DATA 7968,01,0A,0A,02,02,02,0A,0A,0366
1762 DATA 7970,0A,0A,0A,0A,0A,0A,0A,0A,0606
1763 DATA 7978,00,00,00,00,0A,0A,0A,0A,0066
1764 DATA 7980,0A,0A,0A,0A,0A,0A,0A,0A,0606
1765 DATA 7988,0A,0A,0A,0A,0A,0A,01,09,0613
1766 DATA 7990,01,0A,0A,0A,0A,0A,0A,0A,0386
1767 DATA 7998,0A,0A,0A,0A,0A,0A,0A,0A,0606
1768 DATA 79A0,0A,00,00,0A,0A,0A,0A,0A,05C6
1769 DATA 79A8,0A,0A,0A,0A,0A,0A,0A,0A,0606
1770 DATA 79B0,0A,0A,0A,0A,0A,0A,01,09,0613
1771 DATA 79B8,01,0A,0A,0A,0A,0A,0A,0A,0386
1772 DATA 79C0,0A,0A,0A,0A,0A,0A,0A,0A,0606
1773 DATA 79C8,0A,00,01,01,01,01,01,01,053F
1774 DATA 79D0,01,01,01,01,00,01,01,01,00F7
1775 DATA 79D8,01,01,01,01,01,01,01,09,00F7
1776 DATA 79E0,01,01,01,01,01,01,01,01,00FF
1777 DATA 79E8,01,01,00,01,01,01,00,00,00DC
1778 DATA 79F0,00,00,00,00,00,00,00,00,0000
1779 DATA 79F8,00,01,00,00,00,00,00,00,0040
1780 DATA 7A00,00,00,00,00,01,00,00,09,0001
1781 DATA 7A08,00,01,00,00,00,00,00,00,0040
1782 DATA 7A10,00,00,00,00,00,01,00,00,0004
1783 DATA 7A18,00,00,00,00,00,00,00,00,0000
1784 DATA 7A20,00,00,01,01,01,01,01,01,003F
1785 DATA 7A28,01,01,01,00,01,09,01,01,00CF
1786 DATA 7A30,01,01,00,01,01,01,01,01,00DF
1787 DATA 7A38,01,01,01,01,01,01,00,00,00FC
1788 DATA 7A40,00,00,00,00,00,00,00,00,0000
1789 DATA 7A48,00,00,00,01,00,00,00,00,0010
1790 DATA 7A50,00,00,00,00,01,09,00,00,002C
1791 DATA 7A58,01,00,00,00,00,00,00,00,0080
1792 DATA 7A60,00,00,00,00,01,00,00,00,0008
1793 DATA 7A68,00,00,00,00,00,00,00,00,0000
1794 DATA 7A70,00,00,00,00,01,01,00,01,000D
1795 DATA 7A78,01,01,01,01,01,01,01,09,00F7
1796 DATA 7A80,01,01,01,01,01,01,01,01,00FF
1797 DATA 7A88,01,01,01,00,01,00,00,00,00E8
1798 DATA 7A90,00,00,00,00,00,00,00,00,0000
1799 DATA 7A98,00,00,00,00,00,01,00,00.0004
1800 DATA 7AA0,00,00,00,00,00,00,00,09,0009
1801 DATA 7AA8,00,00,00,00,00,00,00,00,0000
1802 DATA 7AB0,00,00,00,00,01,00,00,00,0008
1803 DATA 7AB8,00,00,00,00,00,00,00,00,0000
1804 DATA 7AC0,00,00,00,00,00,00,01,01,0003
1805 DATA 7AC8,01,01,01,01,01,01,01,01,00FF
1806 DATA 7AD0,01,01,01,01,01,01,01,01,00FF
1807 DATA 7AD8,01,01,01,01,00,00,00,00,00F0
1808 DATA 7AE0,00,00,00,00,1A,01,01,01,00D7
1809 DATA 7AE8,01,01,01,01,01,01,01,01,00FF
1810 DATA 7AF0,01,01,01,01,01,01,01,01,00FF
1811 DATA 7AF8,01,01,01,01,01,01,01,01,00FF
1812 DATA 7B00,01,01,01,01,01,01,01,01,00FF
1813 DATA 7B08,01,01,01,01,01,01,09,01,00EF
1814 DATA 7B10,09,09,09,09,09,09,09,09,0707
1815 DATA 7B18,09,02,09,09,09,09,09,09,05C7
1816 DATA 7B20,09,09,09,09,09,09,09,09,0707
1817 DATA 7B28,09,09,01,01,01,09,09,01,06CF
1818 DATA 7B30,09,09,09,09,01,01,09,01,076F
1819 DATA 7B38,09,09,02,09,09,09,09,09,0667
1820 DATA 7B40,01,01,01,01,09,09,09,09,0087
1821 DATA 7B48,09,01,01,01,09,09,09,09,0487
1822 DATA 7B50,09,09,09,09,09,09,09,09,0707
1823 DATA 7B58,01,09,09,09,01,01,09,01,036F
1824 DATA 7B60,01,01,01,01,01,01,09,09,00E7
1825 DATA 7B68,09,02,09,09,09,09,09,09,05C7
1826 DATA 7B70,01,09,01,01,01,09,01,09,02D7
1827 DATA 7B78,09,09,09,09,09,09,09,09,0707
1828 DATA 7B80,01,09,09,09,01,01,09,09,0367
1829 DATA 7B88,01,09,09,09,09,01,09,09,0327
1830 DATA 7B90,01,01,01,09,09,09,09,01,000F
1831 DATA 7B98,02,01,09,01,09,01,09,01,002F
1832 DATA 7BA0,02,09,02,01,01,01,09,09,0307
1833 DATA 7BA8,01,09,02,09,01,01,09,09,0207
1834 DATA 7BB0,09,09,01,02,01,01,09,01,06DF
1835 DATA 7BB8,09,09,09,01,09,09,01,09,0797
1836 DATA 7BC0,01,09,09,01,09,01,09,09,03A7
1837 DATA 7BC8,01,01,09,09,09,09,01,09,0117
1838 DATA 7BD0,01,01,01,09,01,01,09,01,006F
1839 DATA 7BD8,09,09,01,09,09,09,09,01,060F
1840 DATA 7BE0,09,09,09,09,01,01,09,01,076F
1841 DATA 7BE8,09,09,01,01,09,01,09,09,06A7
1842 DATA 7BF0,01,01,09,09,09,09,01,09,0117
1843 DATA 7BF8,09,09,09,01,01,01,09,01,07EF
1844 DATA 7C00,09,09,01,09,09,09,09,01,060F
1845 DATA 7C08,02,01,01,01,01,01,02,01,0179
1846 DATA 7C10,09,09,09,09,09,01,09,09,0727
1847 DATA 7C18,01,01,09,09,09,09,01,09,0117
1848 DATA 7C20,09,09,09,01,01,01,09,01,07EF
1849 DATA 7C28,09,09,09,01,01,09,09,01,07CF
1850 DATA 7C30,01,09,09,09,09,09,01,01,031F
1851 DATA 7C38,09,09,01,01,01,01,09,09,06E7
1852 DATA 7C40,01,09,09,09,09,09,01,09,0317
1853 DATA 7C48,09,09,09,01,01,01,09,02,07EC
1854 DATA 7C50,01,01,01,09,01,09,09,09,0047
1855 DATA 7C58,09,09,09,09,09,09,09,01,070F
1856 DATA 7C60,01,09,01,09,09,09,09,01,020F
1857 DATA 7C68,09,02,01,01,01,01,01,09,0437
```

```
1858 DATA 7C70,01,01,01,09,01,01,09,01,006F
1859 DATA 7C78,09,09,09,09,02,09,09,09,075F
1860 DATA 7C80,01,01,01,09,09,09,09,01,000F
1861 DATA 7C88,09,01,01,09,09,09,01,09,0417
1862 DATA 7C90,09,01,09,09,09,09,09,01,050F
1863 DATA 7C98,09,09,09,09,01,01,09,01,076F
1864 DATA 7CA0,09,01,01,01,01,09,09,01,04CF
1865 DATA 7CA8,09,09,09,01,09,09,09,01,078F
1866 DATA 7CB0,09,01,09,01,09,09,09,09,0587
1867 DATA 7CB8,09,09,01,09,09,09,09,01,060F
1868 DATA 7CC0,09,09,09,09,01,01,09,01,076F
1869 DATA 7CC8,09,01,09,09,09,09,09,01,050F
1870 DATA 7CD0,09,09,09,01,09,09,02,01,0799
1871 DATA 7CD8,09,01,09,09,01,02,09,09,056B
1872 DATA 7CE0,09,09,09,09,09,01,01,09,0737
1873 DATA 7CE8,01,01,01,09,01,01,09,01,006F
1874 DATA 7CF0,09,01,09,09,01,01,09,01,056F
1875 DATA 7CF8,09,09,09,09,01,01,01,09,0777
1876 DATA 7D00,09,01,09,09,09,09,01,09,0517
1877 DATA 7D08,09,09,09,09,09,09,01,09,0717
1878 DATA 7D10,09,09,01,09,01,01,09,09,0667
1879 DATA 7D18,09,09,09,09,09,09,09,09,0707
1880 DATA 7D20,09,09,09,09,09,09,09,01,070F
1881 DATA 7D28,01,01,09,09,09,09,01,09,0117
1882 DATA 7D30,01,01,09,09,09,09,01,09,0117
1883 DATA 7D38,09,09,01,09,01,01,09,09,0667
1884 DATA 7D40,09,09,09,01,01,01,01,01,07FF
1885 DATA 7D48,01,09,09,09,09,09,01,09,0317
1886 DATA 7D50,09,09,09,09,09,09,01,09,0717
1887 DATA 7D58,01,09,09,09,09,09,02,02,031A
1888 DATA 7D60,01,01,09,09,01,01,09,01,016F
1889 DATA 7D68,01,09,02,01,09,09,09,09,02E7
1890 DATA 7D70,01,01,01,01,09,09,01,01,009F
1891 DATA 7D78,01,01,01,02,01,01,01,09,00C7
1892 DATA 7D80,01,09,09,09,09,09,01,01,031F
1893 DATA 7D88,01,01,01,09,01,01,09,01,006F
1894 DATA 7D90,09,09,09,01,09,09,09,09,0787
1895 DATA 7D98,09,09,09,01,01,09,09,09,07C7
1896 DATA 7DA0,09,01,01,01,09,09,09,09,0487
1897 DATA 7DA8,09,01,09,09,09,09,01,09,0517
1898 DATA 7DB0,09,09,01,09,01,01,09,01,066F
1899 DATA 7DB8,09,01,09,01,09,09,01,01,059F
1900 DATA 7DC0,01,01,09,09,01,09,09,09,0147
1901 DATA 7DC8,01,09,09,09,09,09,09,09,0307
1902 DATA 7DD0,09,01,01,01,01,01,09,09,04E7
1903 DATA 7DD8,09,09,01,09,01,01,09,01,066F
1904 DATA 7DE0,09,09,01,01,09,09,01,09,0697
1905 DATA 7DE8,09,01,02,02,01,09,09,09,0497
1906 DATA 7DF0,01,09,09,09,09,09,09,09,0307
1907 DATA 7DF8,09,01,09,09,09,09,09,09,0507
1908 DATA 7E00,09,09,01,09,01,01,02,01,0679
1909 DATA 7E08,09,09,09,09,09,09,01,09,0717
1910 DATA 7E10,09,01,01,01,01,09,02,09,04D1
1911 DATA 7E18,01,01,01,01,01,01,01,01,00FF
1912 DATA 7E20,01,09,09,09,09,09,01,01,031F
1913 DATA 7E28,01,01,01,09,01,01,09,01,006F
1914 DATA 7E30,09,01,09,01,09,09,01,01,059F
1915 DATA 7E38,09,09,09,09,09,09,01,09,0717
1916 DATA 7E40,09,09,09,09,09,09,09,09,0707
1917 DATA 7E48,09,09,09,09,09,09,01,09,0717
1918 DATA 7E50,09,09,09,09,01,01,01,01,077F
1919 DATA 7E58,01,01,01,01,01,01,01,01,00FF
1920 DATA 7E60,01,01,01,01,01,01,01,01,00FF
1921 DATA 7E68,01,01,01,01,01,01,01,01,00FF
1922 DATA 7E70,01,01,01,01,01,01,01,01,00FF
1923 DATA 7E78,01,01,01,01,01,44,A8,17,00AF
1924 DATA 7E80,00,00,0A,0A,0A,0A,0A,0A,0186
1925 DATA 7E88,0A,0A,0A,0A,0A,0A,0A,0A,0606
1926 DATA 7E90,0A,0A,0A,0A,0A,0A,0A,0A,0606
1927 DATA 7E98,0A,0A,0A,0A,0A,0A,0A,0A,0606
1928 DATA 7EA0,0A,0A,0A,00,00,00,00,02,06C2
1929 DATA 7EA8,01,09,00,00,00,00,00,00,02C0
1930 DATA 7EB0,00,00,03,03,03,03,03,03,0041
1931 DATA 7EB8,03,03,03,03,03,03,03,03,0101
1932 DATA 7EC0,03,03,03,03,03,03,03,00,0102
1933 DATA 7EC8,00,01,01,09,01,01,01,01,00FF
1934 DATA 7ED0,00,09,00,00,05,00,00,00,0268
1935 DATA 7ED8,00,00,01,01,01,01,01,01,003F
1936 DATA 7EE0,01,01,01,01,01,01,01,01,00FF
1937 DATA 7EE8,01,01,01,01,01,01,01,00,00FE
1938 DATA 7EF0,00,01,01,09,00,00,00,00,00F0
1939 DATA 7EF8,00,09,00,00,00,00,00,04,0244
1940 DATA 7F00,00,00,00,00,00,00,00,00,0000
1941 DATA 7F08,00,00,00,00,00,00,00,00,0000
1942 DATA 7F10,00,00,00,02,00,00,00,00,0020
1943 DATA 7F18,00,01,01,09,00,00,00,00,00F0
1944 DATA 7F20,01,01,01,01,09,00,00,00,00B8
1945 DATA 7F28,00,00,00,00,00,00,00,00,0000
1946 DATA 7F30,00,00,00,00,00,00,00,00,0000
1947 DATA 7F38,00,00,00,0B,00,00,00,00,00B0
1948 DATA 7F40,00,01,01,01,01,01,09,00,006E
1949 DATA 7F48,00,00,00,00,09,00,00,00,0048
1950 DATA 7F50,00,05,00,00,00,00,00,00,0140
1951 DATA 7F58,00,00,00,00,00,00,00,00,0000
1952 DATA 7F60,09,01,01,0B,01,01,04,00,0454
1953 DATA 7F68,00,01,01,00,00,00,09,00,0072
1954 DATA 7F70,00,00,00,00,09,00,00,00,0048
1955 DATA 7F78,00,00,00,00,00,00,00,00,0000
1956 DATA 7F80,00,00,00,00,00,00,00,00,0000
1957 DATA 7F88,09,01,01,0B,01,01,00,00,045C
1958 DATA 7F90,04,01,01,00,00,00,09,00,0272
1959 DATA 7F98,00,09,01,01,01,01,00,00,027C
1960 DATA 7FA0,00,04,00,00,00,00,00,00,0100
1961 DATA 7FA8,00,00,00,00,00,00,00,00,0000
1962 DATA 7FB0,09,01,01,0B,01,01,05,00,0456
1963 DATA 7FB8,00,01,01,01,01,01,01,09,0077
1964 DATA 7FC0,00,09,00,00,00,00,00,00,0240
1965 DATA 7FC8,00,00,00,00,00,00,00,09,0009
1966 DATA 7FD0,01,01,01,01,01,01,01,01,00FF
1967 DATA 7FD8,01,01,01,0B,01,01,00,00,005C
1968 DATA 7FE0,04,01,01,00,00,00,00,09,0269
1969 DATA 7FE8,00,09,00,00,00,00,00,05,0245
1970 DATA 7FF0,00,00,00,00,00,00,00,09,0009
1971 DATA 7FF8,01,01,01,01,01,01,01,01,00FF
1972 DATA 8000,01,01,01,0B,01,01,05,00,0056
1973 DATA 8008,00,01,01,00,00,00,00,09,0069
1974 DATA 8010,01,01,01,01,09,00,00,00,00B8
1975 DATA 8018,00,00,00,05,00,00,00,09,0059
1976 DATA 8020,01,01,01,01,01,01,01,01,00FF
1977 DATA 8028,01,01,01,0B,01,01,00,00,005C
1978 DATA 8030,04,01,01,09,01,01,01,01,02FF
1979 DATA 8038,00,00,00,00,09,00,00,04,004C
1980 DATA 8040,00,00,00,00,00,00,00,09,0009
1981 DATA 8048,00,00,00,00,00,00,00,00,0000
1982 DATA 8050,00,00,00,0B,00,01,05,00,00BE
1983 DATA 8058,00,01,01,09,00,00,00,00,00F0
1984 DATA 8060,00,00,00,00,09,00,00,00,0048
1985 DATA 8068,00,00,00,00,00,09,01,01,0027
1986 DATA 8070,01,01,01,01,09,01,01,01,00BF
1987 DATA 8078,01,01,01,01,09,01,00,00,00BC
1988 DATA 8080,04,01,01,09,00,00,00,00,02F0
1989 DATA 8088,00,09,01,01,01,01,00,00,027C
1990 DATA 8090,00,00,00,04,00,09,01,00,0066
1991 DATA 8098,00,00,00,00,09,00,00,00,0048
1992 DATA 80A0,00,00,00,01,09,01,05,00,0056
1993 DATA 80A8,00,01,01,01,01,01,09,00,006E
1994 DATA 80B0,00,09,00,00,00,00,00,05,0245
1995 DATA 80B8,00,00,00,00,00,09,01,00,0026
1996 DATA 80C0,00,09,01,01,01,01,01,09,0277
1997 DATA 80C8,00,00,00,01,09,01,00,00,005C
1998 DATA 80D0,04,01,01,00,00,00,09,00,0272
1999 DATA 80D8,00,09,00,00,00,00,00,00,0240
2000 DATA 80E0,00,00,00,00,00,09,01,00,0026
2001 DATA 80E8,00,09,00,00,00,00,00,09,0249
2002 DATA 80F0,00,00,00,01,09,01,05,00,0056
2003 DATA 80F8,00,01,01,00,00,00,09,00,0072
2004 DATA 8100,01,01,01,01,09,00,00,00,00B8
2005 DATA 8108,04,00,00,00,00,09,01,09,022F
2006 DATA 8110,01,01,01,09,00,09,01,01,0057
2007 DATA 8118,01,01,09,01,09,01,00,00,01BC
2008 DATA 8120,04,01,01,09,01,01,01,01,02FF
2009 DATA 8128,00,00,00,00,09,00,00,00,0048
2010 DATA 8130,00,00,00,00,00,09,01,09,002F
2011 DATA 8138,01,01,01,01,01,01,01,01,00FF
2012 DATA 8140,01,01,09,01,09,01,05,00,01B6
2013 DATA 8148,00,01,01,09,00,00,00,00,00F0
2014 DATA 8150,00,00,00,00,09,00,02,00,004C
2015 DATA 8158,00,00,00,00,00,09,01,09,002F
2016 DATA 8160,00,00,01,01,01,01,01,00,003E
2017 DATA 8168,00,00,09,01,09,00,00,00,0178
2018 DATA 8170,04,01,01,09,00,00,00,00,02F0
2019 DATA 8178,00,00,01,01,01,01,01,00,003E
2020 DATA 8180,00,00,00,00,00,09,01,09,002F
2021 DATA 8188,00,00,00,00,00,00,00,00,0000
2022 DATA 8190,00,00,09,01,09,01,05,00,0176
2023 DATA 8198,00,01,01,01,01,09,00,00,005C
2024 DATA 81A0,00,00,00,00,00,00,00,00,0000
2025 DATA 81A8,00,00,00,00,00,09,01,01,0027
2026 DATA 81B0,01,01,01,04,00,05,01,01,00B7
2027 DATA 81B8,01,01,01,01,09,01,01,01,00BF
2028 DATA 81C0,01,01,01,01,01,09,00,00,00DC
2029 DATA 81C8,00,08,08,08,08,08,08,00,03F0
2030 DATA 81D0,00,00,00,00,00,09,01,00,0026
2031 DATA 81D8,00,00,00,01,02,01,00,00,0004
2032 DATA 81E0,00,00,00,01,09,00,00,00,0058
2033 DATA 81E8,00,00,00,00,00,09,00,00,0024
2034 DATA 81F0,01,01,01,01,01,01,01,01,00FF
2035 DATA 81F8,01,01,01,01,01,01,01,00,00FE
2036 DATA 8200,00,00,00,01,00,01,00,00,0014
2037 DATA 8208,00,00,00,01,01,01,01,01,001F
2038 DATA 8210,01,01,01,01,01,01,01,01,00FF
2039 DATA 8218,00,A8,04,2C,30,31,2C,30,296C
```

Listing 4. Fertige Bilder für Stone Runner (Schluß)

Mandelbrots wunderbare Mathematik

Kennen Sie die fantastischen Computerbilder von Mandelbrotberechnungen? Sicher haben Sie sich schon gefragt, wie man so etwas macht. Mit einigen mathematischen Grundlagen und Ihrem CPC ist das ganz einfach.

Sicher haben auch Sie bereits Mandelbrotbilder gesehen. Jene fantastischen farbigen, abstrakt wirkenden Bilder, die wie fotografisch verfremdete Sonneneruptionen aussehen, erschienen wiederholt in der Fachpresse. Die Grundlagen für diese Bilder entwickelte der polnische, in Amerika lebende Mathematiker Benoit B. Mandelbrot.

Um zu verstehen, wie Mandelbrotbilder erzeugt werden, muß man sich mit der Mathematik von Fraktalprogrammen auseinandersetzen. Sie basiert auf Rechnungen mit komplexen Zahlen. Doch keine Angst, mit Ihrem Wissen aus der Schulmathematik und ein paar Hinweisen von uns lernen Sie schnell, mit komplexen Zahlen umzugehen.

Aus der Mathematik sind uns mehrere Zahlenmengen bekannt. Die einfachste ist die Menge der natürlichen Zahlen. Mit ihr kann man ganz normal zählen, also »1, 2, 3...«, und rechnen: »1+1=2«, »4-3=1«. Das funktioniert nur so lange, bis man feststellt, daß »3-4« nicht zu rechnen ist. Man braucht also eine neue Zahlenmenge, mit der das Ergebnis dieser Rechnung darstellbar ist.

Dazu wurde die Menge der ganzen Zahlen eingeführt, die auch negative Zahlen einschließt. Mit dieser Zahlenmenge läßt sich nach Belieben addieren, subtrahieren und multiplizieren. Beim Dividieren hingegen gibt es Fälle, die mit der Menge der ganzen Zahlen auch nicht lösbar sind: »3/2« geht nicht auf. Also wird wieder eine neue Menge erforderlich.

Diese nennt man die Menge der rationalen Zahlen. Sie umfaßt die Menge der ganzen Zahlen einschließlich deren Brüchen. Mit ihr lassen sich fast alle bekannten mathematischen Operationen ausführen.

Die Fläche eines Kreises ist damit jedoch noch immer nicht zu berechnen, da keine der bisher besprochenen mathematischen Mengen die Kreiszahl PI (π) enthält. Eine weitere Menge wird fällig, die Menge der reellen Zahlen.

Innerhalb dieser Menge lassen sich alle mathematischen Operationen bis hin zu Wurzel- und Logarithmusfunktionen ausführen. Allerdings geht das nur solange gut, bis man versucht, die Quadratwurzel aus -1 zu ziehen. Wahrscheinlich kennen Sie das von Ihrem Taschenrechner, der schlicht ein »ERROR« anzeigt, wenn Sie versuchen, aus negativen Zahlen die Quadratwurzel zu bilden. Da die Mathematiker wissen wollten, zu welchem Ergebnis die Wurzel aus -1 führt, ersannen sie eine vorläufig letzte Menge der Zahlen: die Menge der komplexen Zahlen. Genau diese Zahlenmenge ist für die Berechnung der Mandelbrotbilder relevant.

Eine komplexe Zahl besteht immer aus einem Real- und einem Imaginärteil; zusammen ergeben beide die komplexe Zahl: zum Beispiel 3+7i. Dabei gibt die 3 den Wert des Realteils an und 7i den Wert des Imaginärteils. Das kleine »i« kennzeichnet den Imaginärteil. Sie werden sich jetzt sicher fragen, was das Ganze soll. Schließlich haben wir immer noch nicht die Quadratwurzel aus -1 gezogen. Der Trick beim i, und dessen ganzer Zweck ist jedoch, daß i mit sich selbst multipliziert -1 ergibt, umgekehrt also die Quadratwurzel aus -1 eben jenes i bildet. Damit sind nun Wurzeln aus negativen Zahlen zu ziehen.

Oben haben wir aber gesagt, daß jede komplexe Zahl aus je einem Imaginär- und Realteil besteht. Nun, das gilt für i ebenfalls. Ausgeschrieben, so wie wir es oben zeigten, heißt i: 0+1i. Den Realteil läßt man weg, da er in diesem Fall Null ist. Und da i nur einmal gezählt wird, läßt man die Eins ebenfalls weg.

Will man nun mit komplexen Zahlen rechnen, sind einige Regeln zu beachten. Bei der Additition zweier komplexer Zahlen werden Realteile und Imaginärteile gesondert addiert:

```
3+7i+4-9i=7-2i
```

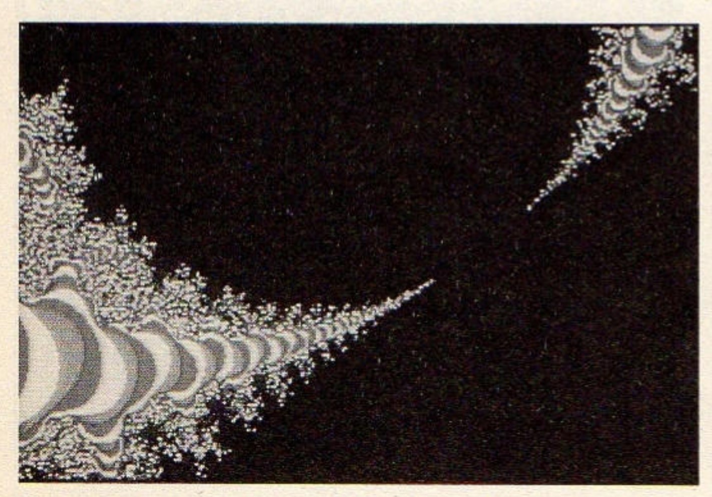
3 plus 4 ergibt 7 und 7 zu -9 addiert ergibt -2, also 7-2i. Entsprechendes gilt bei der Subtraktion komplexer Zahlen.

Die Multiplikation ist schon etwas schwieriger. Will man 3+7i mit 7-2i multiplizieren, geht man folgendermaßen vor: Zuerst wird jedes Element der beiden komplexen Zahlen miteinander multipliziert. Also: 3x7=21, 3x-2i=-6i, 7ix7=49i und 7ix-2i=$-14i^2$. Das ergibt dann

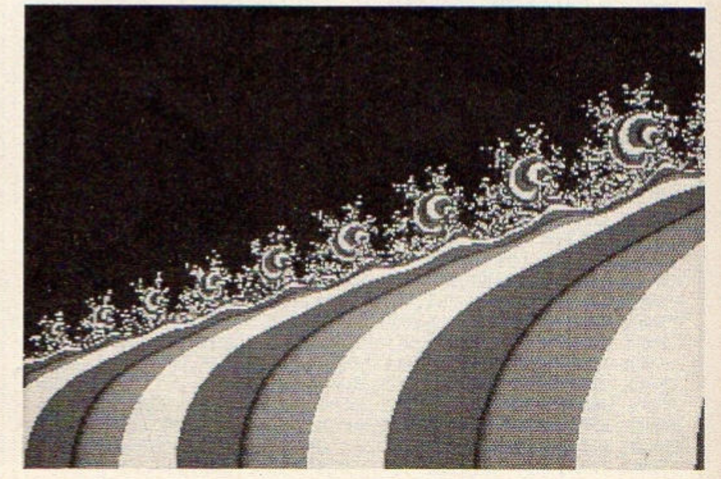
$$21-6i+49i-14i^2$$

Dann wird zusammengefaßt:

$$21+43i-14i^2$$



Erste Tabellenwerte: interessante Konturen



Zweite Tabellenwerte: Bei den Schnörkeln wird's interessant

Da $i^2=-1$ ist, wird aus $-14i^2$ schlicht 14. Das Endergebnis von 3+7i multipliziert mit 7-2i ist demnach 35+43i.

Eine komplexe Zahl läßt sich auch grafisch darstellen. Man handhabt sie wie eine normale Zahl, nur daß auf der x-Achse der Realteil angetragen wird und auf der y-Achse der Imaginärteil. Zeichnet man eine komplexe Zahl in ein Koordinatensystem, so spricht man auch von der »Darstellung« der komplexen Zahl in der »Gaußschen Zahlenebene«.

Die Mandelbrotmenge

Zum Rechnen mit komplexen Zahlen ließe sich noch eine Menge mehr sagen. Um zu verstehen, wie Mandelbrotbilder entstehen, genügt dieses Wissen jedoch.

Die faszinierenden Bilder basieren auf folgender einfachen Formel:

$z=z^2+c$

Die Formel bewirkt nichts anderes, als z zu quadrieren, einen festen Wert c hinzu zu addieren, daraus wieder z zu bilden und so weiter. Wie Sie vielleicht bereits richtig vermuten, sind z und c komplexe Zahlen. Den Vorgang nennt man »iterieren«.

Schön, und wie entstehen daraus Bilder? Ganz einfach: c bildet eine Koordinate in der Gaußschen Zahlenebene. Gleichzeitig ist die Ebene der Bildschirm des Computers. Man nimmt nun jeden Bildpunkt und setzt dessen Koordinate in die Gleichung als c ein. Das Merkwürdige an dieser Gleichung ist nun, daß z entweder sehr schnell unendlich groß wird oder aber sehr langsam. Man zählt nun ganz einfach mit, nach wievielen Durchläufen z unendlich groß geworden ist. Diese Zahl bestimmt dann eine Farbe, wobei gleiche Zahlen für gleiche Farben stehen, und färbt damit den entsprechenden Bildpunkt auf dem Bildschirm ein. Indem man diesen Vorgang mit allen Bildschirmpunkten wiederholt, erhält man die Mandelbrotbilder.

Wir brauchen jetzt bloß noch dem Computer den Umgang mit dieser Gleichung beizubringen. Betrachten wir dazu der Einfachheit halber erst einmal den Teil rechts vom Gleichheitszeichen. Dort steht: z^2+c, wobei z die komplexe Variable und c die Koordinate als komplexe Zahl darstellt. Da der Computer nicht mit komplexen Zahlen rechnen kann, müssen wir die Formel für ihn aufbereiten. Beginnen wir also mit z. Da z für eine komplexe Zahl steht, ist z auch in je einen Imaginär- und Realteil aufzuspalten. Das ergibt dann a+bi. In der Formel ist z quadriert, die aufgespaltene Schreibweise würde dann also folgendermaßen aussehen:

$z^2=(a+bi)^2$

$z^2=a^2+2abi+bi^2$

Wir haben aber gelernt, daß i^2 den Wert -1 hat, dann heißt es also:

$z^2=a^2-b^2+2abi$

Ebenso läßt sich c aufspalten, um d+ei zu erhalten. Wenn wir nun die beiden komplexen Zahlen z und c aus der Originalgleichung zusammenfassen, und nach Real- und Imaginärteilen sortieren, erhalten wir:

$z=a^2-b^2+d+2abi+ei$

Für den Computer spaltet man die Gleichung weiter in Real- und Imaginärteil auf, die dann getrennt ausgerechnet werden. Das z links vom Doppelpunkt ist eine komplexe Variable a+bi. Wenn man a+bi als zwei separate Gleichungen auffaßt, in der alle Real- und Imaginärteile je eine Gleichung bilden, so erhält man:

$a=a^2-b^2+d$

und

$bi=2abi+ei$

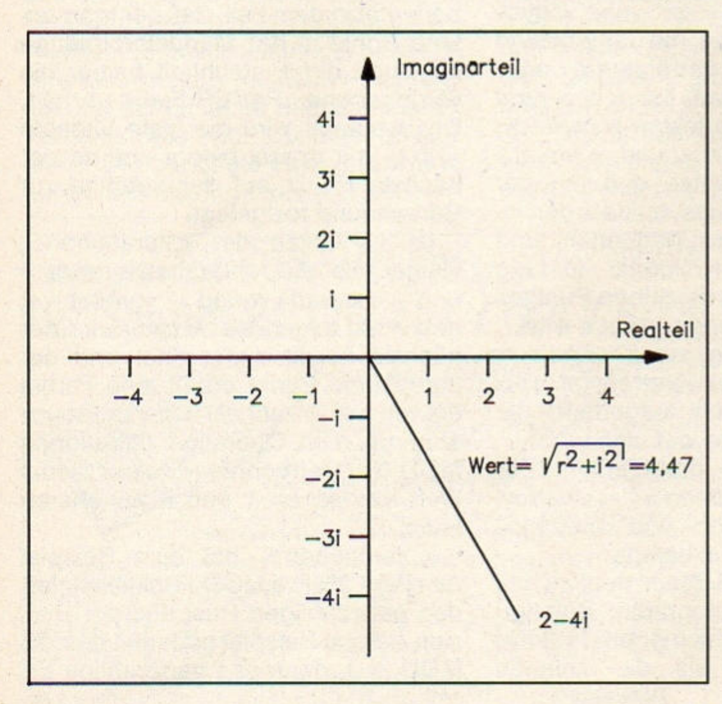
Für den Computer nimmt man für die imaginäre Zahlen normale reale Zahlen an und erhält:

$a=a^2-b^2+d$

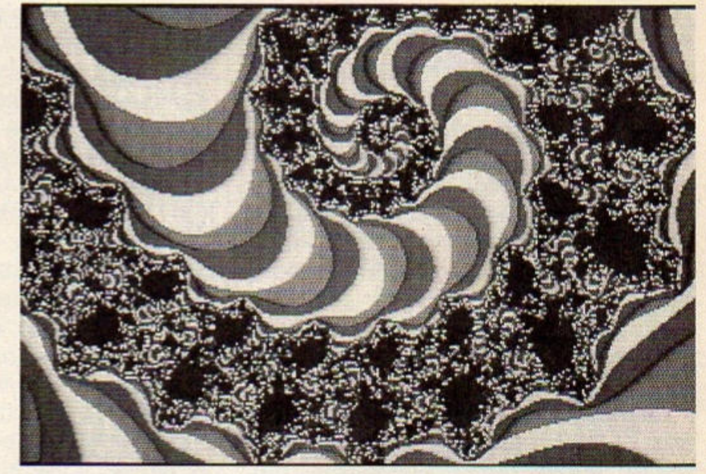
$b=2ab+e$

Damit kann der Computer nun rechnen.

Wie bereits erwähnt, wird z in der Gleichung entweder sehr schnell unendlich groß oder aber sehr langsam. Man überprüft nun nicht, ob die Zahl unendlich groß ist, denn das ist auf dem Computer schwer realisierbar. Vielmehr testet man, ob der Wert der komplexen Zahl z größer als 2 ist. Der Wert der komplexen Zahl ist wie folgt zu berechnen. Oben haben wir gesehen, daß sich eine komplexe Zahl wie normale Koordinaten in ein Koordinatensystem einzeichnen läßt. Der Wert der komplexen Zahl ergibt sich aus dem Abstand der Koordinate zum Nullpunkt des Koordinatensystems. Dazu werden die Imaginär- und Realteile jeweils separat quadriert und die Quadrate addiert. Beim Quadrieren wird das i (der Imaginärteil) nicht beachtet. Aus dem Ergebnis zieht man die Wurzel und erhält so den Wert der komplexen Zahl.



So bilden Real- und Imaginärteile komplexer Zahlen die Bildschirmkoordinaten



Dritte Tabellenwerte: Eine endlose Treppe

Das Mandelbrotprogramm

Nach soviel Grundlagen und Theorie wollen wir uns nun endlich der Problemlösung und dem Programm widmen. Das Programm wurde auf dem Schneider CPC 464 unter Verwendung des Hisoft-Pascalcompilers

geschrieben. Dabei ist folgendes zu beachten:

a) Im Programm werden Compiler-Systembefehle benutzt, die nicht zum Standardpascal gehören. Die Anpassung an beispielsweise Turbo-Pascal fällt aber nicht allzu schwer.

b) Nachdem Sie den Compiler mit »RUN"PASCAL"« gestartet haben, müssen Sie auf die Frage »RAM-Top (ENTER for default)?« mit »&9FFF« antworten, sonst funktioniert die Eingabe des Dateinamen nicht korrekt.

c) Listing 1 und 2 des Programms sind getrennt einzugeben. Den Prozedurenteil speichern Sie mit »P <Nummer der ersten Zeile>, <Nummer der letzten Zeile>,MANDEL.LIB«, den Hauptteil mit »P <Nummer der ersten Zeile>, <Nummer der letzten Zeile>,MANDEL.PAS«.

d) Wenn das abgetippte Programm fehlerfrei läuft, läßt sich eine Binärversion compilieren. Das geschieht folgendermaßen: Zuerst stellen Sie sicher, daß das Hauptprogramm im Arbeitsspeicher steht und die Diskette mit den Prozeduren sich im Laufwerk befindet. Dann geben Sie im Editor »F1,2,<Programmname>« ein. Nach der Eingabe von »T <Nummer der ersten Zeile>« auf den Prompt »OK« mit <Y> antworten. Die Binärversion wird nun unter dem gewählten Namen auf Diskette gespeichert.

e) In beiden Programmteilen steht als erste Zeile der Befehl {$L-}, zur Unterdrückung der Listingausgabe auf dem Bildschirm während des Compiliervorgangs.

Bevor das Programm etwas berechnet, ist die Eingabe einiger Startparameter notwendig. Es erwartet der Reihe nach folgende Eingaben:

1. ITERATIONEN: Der Wert bestimmt, nach wieviel Rechendurchläufen das Programm spätestens mit dem Wiederholen der Rechenschleife aufhören soll. Die Variable <iterationsgrenze> wird mit diesem Wert belegt.

2. XMIN: Die Eingabe des kleinsten x-Werts im Intervall von (-2.5 bis 0.75).

3. XMAX: Die Eingabe des größten x-Werts im gleichen Intervall.

4. YMIN: Die Eingabe des kleinsten y-Werts im Intervall von (-1.5 bis 1.5).

5. YMAX: Die Eingabe des größten y-Werts im y-Intervall.

6. FILENAME: Hier erfolgt die Eingabe des Dateinamens für das errechnete Bild. Der Dateiname darf maximal acht Buchstaben lang sein.

Wie lang braucht der Computer?

Im Programm folgt nun eine Initialisierung des Bildschirms, damit alle Bildschirmparameter auf den Einschaltzustand zurückgesetzt sind. Ebenso wird eine Uhr installiert, da es interessant zu wissen ist, wieviel Zeit der Computer zur Berechnung eines Ausschnitts aus der Mandelbrotmenge benötigte. <dx> und <dy> sind reelle Variablen, die den Abstand zweier Pixel in den gewählten x- und y-Intervallen enthalten. Mit <cx> und <cy> werden die festen komplexen Zahlen mit <xmin> und <ymin> belegt. Das bedeutet, daß mit der Berechnung des Ausschnitts in der linken unteren Ecke begonnen und gleichzeitig ein Nullpunkt für die Berechnung jedes einzelnen Punktes der Mandelbrotmenge gesetzt wird.

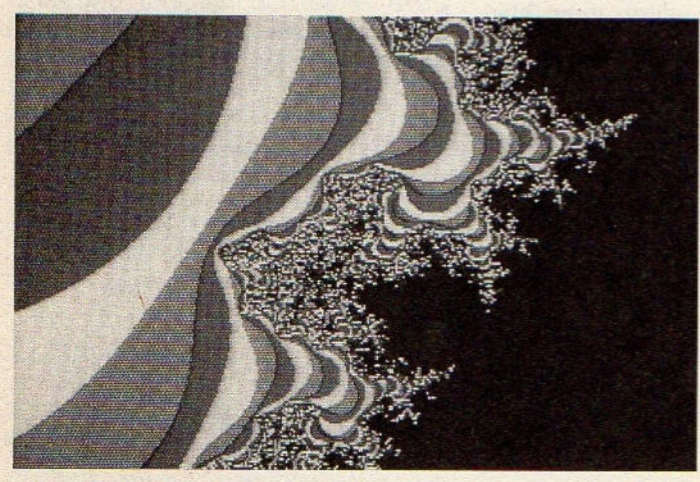
Zur Berechnung jedes einzelnen Punktes der Mandelbrotmenge werden zwei Schleifen aufgebaut, die jeweils alle Punkte auf der y-Achse und der x-Achse durchlaufen. Die Variablen <iterationen>, <xwert>, <ywert>, <xquad> und <yquad> werden alle mit Null belegt.

Mit der WHILE-Schleife beginnt das eigentliche Hauptprogramm. Solange die bereits durchlaufenen <iterationen> kleiner als die anfangs gesetzte <iterationsgrenze> und die Summe aus <xquad> und <yquad> kleiner als 4 ist, soll mit der Berechnung der Mandelbrotmenge fortgefahren werden.

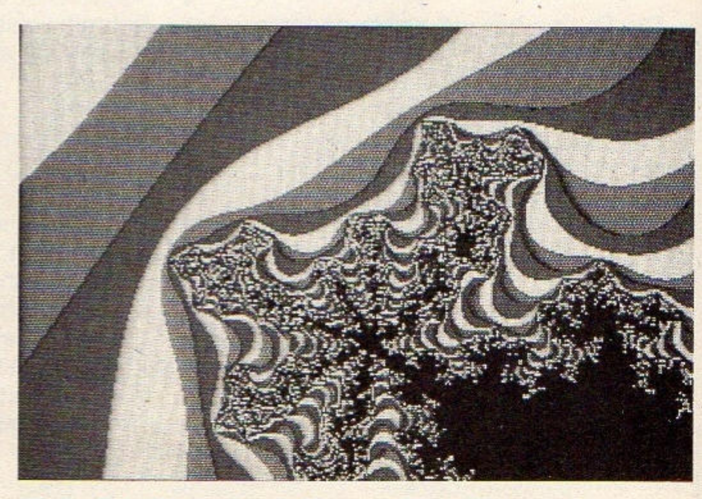
Die Mandelbrotmenge berechnet sich wie folgt: Der Imaginärteil <ywert> der Gleichung wird aus 2x<xwert>x<ywert>+cy berechnet. Den Realteil <xwert> berechnet xquad-yquad+cx, <xquad> und <yquad> enthalten jeweils die Quadrate von <xwert> und <ywert>. Als letzter Schritt in der Mandelbrotmengenberechnung wird <iterationen> um 1 erhöht. Bricht die WHILE-Schleife nun die Berechnungsroutine ab, das heißt <iterationen> = <iterationsgrenze> oder <xquad+yquad> =4, wird dem aktuellen Punkt der Mandelbrotmenge eine bestimmte Farbe zugeordnet. War die Anzahl der <iterationen> gleich der angegebenen <iterationsgrenze>, wird dem Punkt keine Farbe zugeordnet. Keine Farbe bedeutet in dem Fall, daß der momentane Punkt in der Mandelbrotmenge liegt und der Einfachheit halber die vorgegebene PAPER-Farbe erhält. Des weiteren wird die feste Variable <cx> mit cx+dx belegt und so der nächste Punkt auf der x-Achse zur Berechnung festgelegt.

Ist die Anzahl der <iterationen> kleiner als die <iterationsgrenze> und <xquad+yquad> größer 4, geschieht folgendes: <cx> wird der nächste Punkt zugeordnet und der berechnete Punkt erhält eine Farbe, die von der Anzahl der <iterationen> abhängt. Die Operation (iterationen MOD 3)+1 errechnet <colour>, die Werte zwischen 1 und 4 annehmen kann.

<iterationen> hat zum Beispiel den Wert 25. Die MOD-Funktion liefert den ganzzahligen Rest aus der Division A/B. Im Beispiel bedeutet das: 25 MOD 3=1, denn das ganzzahlige Er-



Vierte Tabellenwerte: Norwegens Fjorde?



Fünfte Tabellenwerte: Je kleiner, desto filigraner

Iterationen:	150	150	150	75	50
XMIN:	-0,55	0,26	-0,7459	-0,798	-0,748
XMAX:	-0,45	0,27	-0,7445	-0,6	-0,6
YMIN:	0,5	0,00000	0,1132	0,2925	0,43
YMAX:	0,5625	0,00625	0,112325	0,435	0,5225

gebnis aus 25/3 ist 8, Rest 1. <colour> erhält also den Wert 2. Der Prozedur <pen> wird der Wert von <colour> übergeben und die Prozedur <plot> zeichnet einen Punkt an die momentane Position. Dazu müssen die Werte von <spalte> und <zeile> verdoppelt werden, da sich beim CPC ja nur »halbe Pixel« ansteuern lassen. Dieser Vorgang wird solange wiederholt, bis die <spalte>-Schleife ihr Ende erreicht hat und alle X-Werte in dieser Spalte berechnet sind.

Dann wird der festen komplexen Zahl <cx> wieder der Anfangspunkt auf der x-Achse (XMIN) zugewiesen. <cy> wird um einen weiteren Punkt auf der y-Achse erhöht, das Ganze beginnt nun wieder von neuem. Ist die <zeile>-Schleife ebenfalls beendet, wird die Prozedur »scnsave« aufgerufen und der gesamte Bildschirminhalt als Binärdatei auf Diskette gespeichert. Die benötigte Rechenzeit wird auf dem Bildschirm ausgegeben.

Einige Hinweise noch zur Bedienung des Programms: Die Farben für die Mandelbrotausschnitte wurden so gewählt, daß bei völlig herabgeregelter Helligkeit des Monitors der Bildschirm fast schwarz ist und damit ein Einbrennen von Punkten in den Bildschirm verhindert wird. Bei der Eingabe des Dateinamens am Programmanfang ist folgendes zu beachten: Eingabefehler sind nicht mit der <DEL>-Taste zu löschen. Ist der Dateiname kürzer als acht Buchstaben, ist er mit Leerzeichen aufzufüllen.

Das Seitenverhältnis von X-Intervall zu Y-Intervall der Koordinaten sollte 1,6 zu 1 betragen, da der Mandelbrotmengenausschnitt sonst verzerrt wird. Im Beispiel: XMIN=-3, XMAX=1 und YMIN=-1,25. Das x-Intervall ist vier Einheiten groß. Das y-Intervall muß nun 4/1,6=2,5 Einheiten groß werden, damit der Ausschnitt nicht verzerrt erscheint. Die Koordinaten YMAX=YMIN+2,5=1,25 schließen das Mandelbrot ein. Eine <iterationsgrenze> von 50 genügt bereits, um das Apfelmännchen sauber auf dem Bildschirm zu zeigen. Je kleiner Sie die Ausschnitte aus der Mandelbrotmenge wählen, um so größer muß die Iterationsgrenze gewählt werden. Zum Abschluß finden Sie in der Tabelle einige interessante Plätze der Mandelbrotmenge. So bleibt uns nur noch, Ihnen viel Spaß im Reich des mathematishen Chaos zu wünschen.

(Bernd Baldin/hf/ma/ja)

```
{$L-}
PROGRAM Mandelbrotmenge;
TYPE Filename=ARRAY[1..12]
              OF Char;
VAR
  Iterationen,
  Iterationsgrenze,
  Zeile,
  Spalte,
  Sec, Min, Hour,
  Colour               : Integer;
  xmin, xmax,
  ymin, ymax,
  xwert, ywert,
  xquad, yquad,
  dx, dy,
```

Listing 1. (Fortsetzung auf Seite 144)

```
    cx, cy              : Real;
    File                : Filename;

  {$F MANDEL  .LIB}

  BEGIN
    Mode(1);
    Ink(3,1,1);Ink(2,3,3);
    Ink(1,13,13);Ink(0,0,0);
    Border (0,0);
    Write('Iterationen ->');
    Read(Iterationsgrenze);
    Write('xmin        ->');
    Read(xmin);
    Write('xmax        ->');
    Read(xmax);
    Write('ymin        ->');
    Read(ymin);
    Write('ymax        ->');
    Read(ymax);
    Write('Filename    ->');
    InLine(#06,#08,#21,  00);
    InLine(#A0,#CD,#06,#BB);
    InLine(#77,#7E,#CD,#5A);
    InLine(#BB,#23,#10,#F4);
    Mode(1);
      Sec:=0; Min:=0; Hour:=0;
      Every(50,1,Clock);
      dx:=(xmax-xmin)/320;
      dy:=(ymax-ymin)/200;
      cx:=xmin;
      cy:=ymin;
      FOR Zeile:=0 TO 199 DO
      BEGIN
        FOR Spalte:=0 TO 319 DO
        BEGIN
          Iterationen:=0;
          xwert:=0; ywert:=0;
          xquad:=0; yquad:=0;
          WHILE (Iterationen
                 > Iterationsgrenze)
          AND (xquad+yquad > 4) DO
          BEGIN
            ywert:=2*xwert*ywert+cy;
            xwert:=xquad-yquad+cx;
            xquad:=Sqr(xwert);
            yquad:=Sqr(ywert);
            Iterationen:=Succ(Iterationen);
          END;

            IF Iterationen=Iterationsgrenze
            THEN cx:=cx+dx
            ELSE
            BEGIN
              cx:=cx+dx;
              Colour:=(Iterationen MOD 3)+1;
              Pen(Colour);
              Plot(Spalte*2 , Zeile*2);
            END;
          END;
          cx:=xmin;
          cy:=cy+dy;
        END;
        Scnsave;
        WriteLn(Hour:2,':',Min:2,':',Sec:2);
      END.
```

Listing 1. Ein relativ kurzes Pascal-Programm führt Sie in die fantastische Bilderwelt der »Apfelmännchen«

```
PROCEDURE Mode (x : Integer);
BEGIN
  Write(Chr(4),Chr(x));
END;

PROCEDURE Pen (Pen : Integer);
BEGIN
  RA:=Chr(Pen);
  User(#BBDE);
END;

PROCEDURE Ink
(Inknr,Colour1,Colour2 : Integer);
BEGIN
  RA:=Chr(Inknr);
  RB:=Chr(Colour1);
  RC:=Chr(Colour2);
  User(#BC32);
END;

PROCEDURE Plot (x,y : Integer);
BEGIN
  RDE:=x;
  RHL:=y;
  User(#BBEA);
END;

PROCEDURE Scnsave;
BEGIN
  InLine(#CD,#65,#BC,#21,#00,#A0,#11);
  InLine(#FF,#A0,#06,#08,#CD,#8C,#BC);
  InLine(#21,#00,#C0,#11,#00,#40,#3E);
  InLine(#02,#CD,#98,#BC,#CD,#8F,#BC);
END;


PROCEDURE Border
(Colour1,Colour2 : Integer);
BEGIN
  RB:=Chr(Colour1);
  RC:=Chr(Colour2);
  User (#BC38);
END;

PROCEDURE Clock;
BEGIN
  Sec:=Succ(Sec);
  IF Sec=60 THEN
  BEGIN
    Sec:=0;
    Min:=Succ(Min);
    IF Min=60 THEN
    BEGIN
      Min:=0;
      Hour:=Succ(Hour);
    END;
  END;
END;
```

Listing 2. Die Prozedur-Bibliothek enthält wichtige Routinen zur Darstellung der Mandelbrotmenge

Fehlermeldungen in Assembler

Eine Assemblerroutine, in ein Basic-Programm eingebunden, beschleunigt den Programmablauf. Doch was tun, wenn die Routine nicht richtig bedient wird? Fehlermeldungen helfen.

Das eingebaute Basic des CPC in allen Ehren, doch manchmal kommt man nicht darum herum, Routinen in Maschinensprache zu programmieren.

Der Aufruf einer Maschinensprachroutine geschieht vom Basic aus bekanntlich mit der Anweisung »CALL«. Nach dem CALL-Befehl wird die Adresse angegeben, an der die Routine steht und eventuell noch mehrere Parameter, die das Maschinenprogramm steuern oder von ihm bearbeitet werden sollen.

Bei der Übergabe der Parameter tauchen aber sehr leicht Fehler auf, zum Beispiel durch einen falschen Parameter oder durch zu viele oder zu wenige Werte hinter dem CALL-Befehl. Wenn die Maschinencode-Routine darauf nicht vorbereitet ist, »hängt« sich der Computer oft auf. Wenn man als Programmierer auf solche Umstände geachtet hat, bleibt einem eigentlich nur die Möglichkeit, die Routine wieder zu verlassen. Der Computer meldet sich dann unverrichteter Dinge zurück, und man hat keinen blassen Schimmer, was schiefgelaufen ist. Wie schön wäre es, wenn man in dem Fall vom Maschinenprogramm aus eine Fehlermeldung produzieren könnte.

Der einfachste Weg dazu ist der, die eingebaute Maschinenroutine des Basic-Interpreters zu benutzen. Ein Beispielprogramm, das nichts anderes macht als einen »Syntax Error« zu produzieren, sieht folgendermaßen aus:

```
      LD    E,2
      RST   18H
      DEFW  ADR
ADR:  DEFW  0CA94H
      DEFB  0FEH
```

Der Fehlerroutine im ROM muß der Code des zu meldenden Fehlers als Zahl im E-Register (CPC 664 und 6128: A-Register) übergeben werden. Die 2 im Programm bedeutet also »Syntax Error«, eine 5 stünde für »Improper Argument« und eine 22 für »Operand missing«. Sämtliche Fehlermeldungen sind im CPC-Handbuch aufgelistet.

Mit »RST 18H« veranlaßt man den Computer, eine beliebige Speicherstelle im RAM oder ROM aufzurufen. Dem »RST«-Befehl folgt dafür ein Zeiger, der auf die Adresse der Routine weist. Im Programmbeispiel ist das der Befehl »DEFW ADR«. An der Stelle, auf die der Zeiger weist, steht nun zusätzlich nach der Adresse der aufzurufenden Routine ein Statusbyte. Dieses Byte bestimmt, ob sich die entsprechende Adresse im Speicher ROM oder RAM befindet. Der Befehl »DEFW CA94H« im Beispielprogramm gibt nun die Adresse der Fehlerroutine an (CPC 664: CB58 hex, CPC 6128: CB55 hex). Das Statusbyte schaltet in diesem Fall das ROM im oberen Speicherbereich zwischen C000 hex und FFFF hex ein.

Will man nun eine Maschinenroutine zur Unterstützung eines Basic-Programms schreiben, und soll diese Werte vom Basic-Programm übernehmen, so muß man sich anschauen, welche Hilfsmittel der Basic-Interpreter dazu zur Verfügung stellt. Nach Ausführung des CALL-Befehls steht im Akku (A-Register) die Anzahl der übergebenen Parameter. Als erstes muß das Maschinenprogramm überprüfen, ob auch alle Parameter im Basic-Programm angegeben sind. Nehmen wir an, das Maschinenprogramm erwartet einen Parameter. Dann müssen die ersten Zeilen der Routine so aussehen:

```
CP    01
JP    NZ,ERROR1
```

Damit wird die Routine »ERROR1« aufgerufen, wenn eine andere Anzahl von Parametern übergeben wurde.

Als nächstes muß sich die Routine den vom Basic-Programm übergebenen Wert holen. Im IX-Register steht dazu als Zeiger eine Adresse. Da Werte in einem Bereich zwischen -32768 und +32767 bei vorzeichenbehafteten Zahlen oder zwischen 0 und 65535 bei vorzeichenlosen Zahlen übergeben werden dürfen, besteht der Wert aus zwei Byte. Beim Z80 steht das niederwertige Byte einer 16-Bit Zahl immer vor dem höherwertigen Byte, so daß die Adresse im IX-Register auf das Low-Byte des Werts zeigt.

Nehmen wir nun an, daß der zu übergebende Wert zwischen 0 und 26 liegen soll. Der folgende Programmteil muß nun also die Einhaltung dieser Grenzen kontrollieren. Programmtechnisch ist es einfacher, erst das High-Byte zu testen, denn wenn dieses größer als 0 ist, war der übergebene Wert größer als 255. Dann muß zum IX-Register eine 1 hinzugezählt werden, damit es auf das High-Byte zeigt.

```
LD    A,(IX+01)
CP    00
JP    NZ,ERROR2
```

Liegt der übergebene Wert also über 255, wird die Routine »ERROR2« aufgerufen. Da hier ein anderer Fehler als im vorigen Programmteil abgefangen wird, wird auch eine andere Fehlerroutine benutzt, die eine andere Fehlermeldung ausgibt.

Als nächstes wird das Low-Byte des übergebenen Wertes auf die Grenzen zwischen 0 und 26 getestet.

```
LD    A,(IX+00)
CP    00
JP    C,ERROR2
CP    27
JP    NC,ERROR2
```

Damit ist sichergestellt, daß der Wert zwischen 0 und 26 liegt. Um das Programm nicht nutzlos im Raum stehen zu lassen, soll es die Bildschirmrandfarbe umschalten.

```
LD    B,A
LD    C,A
CALL  BC38H
RET
```

Die Betriebssystemroutine bei der Speicheradresse BC38 hex schaltet die Randfarbe um. Im B- und C-Register stehen dabei die beiden Farben, zwischen denen der Computer hin- und herwechseln soll. Wenn der Rand nicht blinken soll, müssen also beide Register den gleichen Wert enthalten. Am Schluß des Programms stehen die beiden Fehlerroutinen.

```
ERROR1: LD    E,2
        JP    ERROR3
ERROR2: LD    E,5
ERROR3: RST   18H
        DEFW  ADR
ADR:    DEFW  0CA94H
        DEFB  0FEH
        RET
```

Damit haben Sie ein komplettes kleines Demonstrationsprogramm, das die Randfarbe umschaltet und darauf achtet, daß beim CALL-Befehl ein Wert übergeben wird, der im Wertebereich zwischen 0 und 26 liegt. Die so erzeugten Fehlermeldungen unterscheiden sich nicht von denen des Basic-Interpreters.

(Jörg Braun/hf)

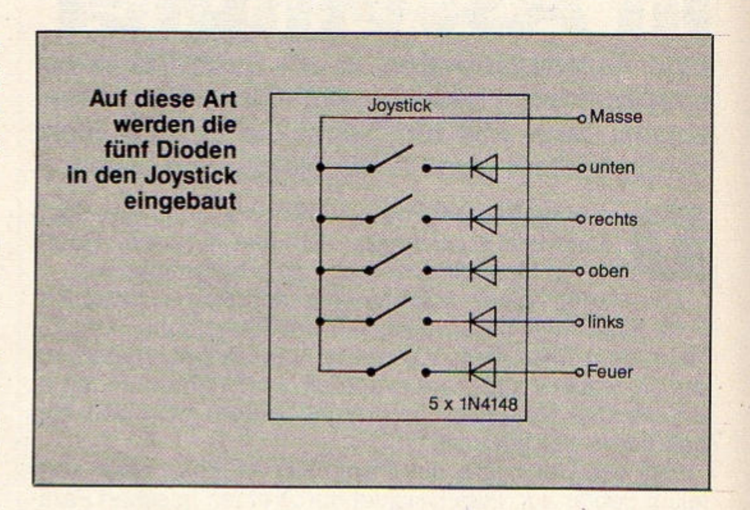
Joystick-Ärger schnell behoben

Spiele, die den Einsatz von zwei Joysticks gleichzeitig erlauben, funktionieren manchmal nicht richtig, wenn Sie nicht spezielle Schneider-Joysticks, sondern die eines Fremdherstellers beim Spielen verwenden.

Haben Sie sich auch schon darüber geärgert, daß Spiele wie »Gauntlet«, »Top Gun« oder »Ikari Warriors« im 2-Spieler-Modus nicht korrekt funktionieren, wenn Sie mit zwei Joysticks gleichzeitig spielen?

Dies liegt nicht etwa am Spielprogramm, sondern an der eigenwilligen Art der Joystickabfrage, die der Schneider CPC benutzt. Die Joysticks von Schneider sind für die Abfragemethode des CPC ausgerüstet, doch bei den wenigsten Joysticks der anderen Hersteller ist dies der Fall, so daß deren Geräte im oben genannten Fall Fehlfunktionen liefern.

Jedoch läßt sich auch der Joystick eines Fremdherstellers leicht zu einem »Schneider-kompatiblen« Joystick umrüsten. Sie müssen lediglich in die fünf Leitungen, die zu den Joystick-Schaltern führen, je eine Standarddiode vom Typ 1N 4148 einlöten. Alle Dioden müssen dabei in Richtung des Joystickschalters gepolt sein (Bild).



Auf diese Art werden die fünf Dioden in den Joystick eingebaut

Für den Einbau der Dioden finden Sie sicherlich im Gehäuse des Joysticks genügend Platz. Die Dioden selbst sind überall im Fachhandel erhältlich (der Hobbybastler hat davon ohnehin einen kleinen Vorrat) und kosten zusammen unter einer Mark, so daß für Bastler der Umbau dem Kauf eines neuen Joysticks allemal vorzuziehen ist.

(Robert Grau/ma)

Verbessertes PIP

Das Programm »PIP« unter CP/M verhält sich höchst unfreundlich, wenn man beim Kopieren die Diskette wechselt. Man muß das Programm dann von neuem starten. Eine kleine Änderung löst dieses Problem.

Sicher ist es Ihnen schon passiert, daß Sie unter CP/M mit »PIP« Dateien kopieren wollten und dabei zwischendurch eine neue Diskette ins Laufwerk einlegten, ohne zur Initialisierung <CTRL+C> gedrückt zu haben. CP/M meldet sich dann unerfreulicherweise mit einem »BDOS ERROR R/O«.

Das Programm PIP ist aber ganz leicht so abzuändern, daß es selber merkt, wenn eine neue Diskette eingelegt wurde. Dazu laden Sie aus dem CP/M das Programm DDT. Sie geben also ein:

```
DDT PIP.COM <ENTER>
```

Der Debugger meldet sich mit:

```
DDT VER 2.2
NEXT    PC
1E00    0100
```

Geben Sie nun »S0100 <ENTER>« ein. Auf dem Bildschirm erscheint:

```
0100    C3
```

Drücken Sie einmal <ENTER>, worauf auf dem Bildschirm eine weitere Zeile erscheint. Geben Sie dann »B2 <ENTER>«, »1D <ENTER>« und ». <ENTER>« ein.

Der Bildschirm muß anschließend so aussehen:

```
0100    C3
0101    CE    B2
0102    04    1D
0103    C9    .
```

Als nächstes geben Sie »S1DB2 <ENTER>« ein, gefolgt von diesen Werten:

```
1DB2    00    0E    <ENTER>
1DB3    00    0D    <ENTER>
1DB4    00    CD    <ENTER>
1DB5    00    05    <ENTER>
1DB6    00    <ENTER>
1DB7    00    C3    <ENTER>
1DB8    00    CE    <ENTER>
1DB9    00    04    <ENTER>
1DBA    00    .     <ENTER>
```

Sie haben jetzt das Programm PIP abgeändert. Um dies auf Diskette zu speichern, gehen Sie mit <CTRL+C> zurück ins CP/M und geben »SAVE 29 PPIP.COM« ein. Auf Ihrer Diskette befindet sich jetzt die abgeänderte Version des PIP, die Sie mit »PPIP« aufrufen.

Für den Assembler-Kundigen die ganze Routine in Z80-Quellcode:

```
0100    JP    1DB2
1DB2    LD    C,0D
1DB4    CALL  0005
1DB7    JP    04CE
```

(Werner Bandorf/hf)

PRINT USING de Luxe

In fast allen Basic-Dialekten existiert zwar der PRINT-USING-Befehl, doch wird er meist sehr sparsam eingesetzt. Gerade beim Zusammenstellen langer Listen ist dieser Befehl jedoch sehr praktisch, weil er Texte und Zahlen auf dem Bildschirm und Drucker exakt ausrichtet.

Der PRINT-USING-Befehl bietet dem Anwender die Fähigkeit, 80 und mehr Zeichen lange Formatmasken zu erzeugen, die sich mit Variablenwerten füllen lassen. Ein Beispiel verdeutlicht dies:

Es soll eine Verkaufsliste formatiert ausgegeben werden. Weitere Bedingung ist, daß bei Dezimalstellen der in der amerikanischen Schreibweise übliche Punkt durch das Komma ersetzt wird.

```
10 MODE 2
20 stueck=6:einheit$="Pfd":lfdnr=133
30 Bez$="Kaeseloecher":ep=2.56
40 komma$=string$(3,8)+","+string$(2,9)
50 zeile$=" --- Nr. #### = ### \  \ \"
   +SPACE$(20)+"\ EP ###.##"+komma$+" DM  GP
   ###.##"+komma$+" DM ---"
60 PRINT USING zeile$;lfdnr,stueck,einheit$,bez$,
   ep,stueck*ep
```

Wenn Sie sich das Ergebnis anschauen, sehen Sie, daß der Ausdruck korrekt mit einem Komma bei Dezimalbrüchen erfolgt.

Es ist auch möglich, die Steuerzeichen von 0 bis 31 sowie TAB-Befehle im Format unterzubringen. Dadurch kann man einzelne Zeilenteile andersfarbig hervorheben, den Cursor an verschiedene Bildschirmpositionen bewegen, den Klingelton integrieren, inverse Zeichen ausgeben und vieles mehr.

Stehen in komplett gezeichneten Kästchen Ihre Punkte- und sonstige Anzeigen, dann lassen sich mit einem einzigen Formatstring, der die entsprechenden Cursorsteuerzeichen enthält, sämtliche Werte auf einen Schlag ausgeben.

Falls Sie noch Bedarf für neue Lotto-Zahlen haben, hier ein kleiner Einzeiler, der Ihnen über den PRINT-USING-Befehl hübsch formatiert zehn Zufallszahlenreihen präsentiert.

```
10 CLS:PEN 3:PRINT"LOTTO 6 aus 49":PRINT STRING$
   (14,218):PEN 1:DIM z(9,49):FOR a=0 TO 9:FOR b=
   1 TO 6:z=0:WHILEz(a,z)=z:z=INT(RND*49)+1:WEND:
   z(a,z)=z:LOCATE 4*a+2,b*2+4:PRINT USING"##";
   z:NEXT b,a:PRINT:PEN 2:PRINT:PRINT"Weitermit
   Taste":CALL &BB18:RUN
```

(Dietmar Schulze/ma)

Vertauschte Zeichensätze unter CP/M

Ein Programm wie »Language« ist überflüssig. Zeichensätze lassen sich bei CP/M Plus auch mit einfachen Steuersequenzen von CCP aus über die Tastatur tauschen.

Leider wird von Schneider für die Diskettenstation das 3-Zoll-Format favorisiert. Da nicht viele Hersteller dieses Format anbieten, können die wenigen die Preise diktieren, und man ist gezwungen, viel Geld für seine Disketten auszugeben. Es sei denn, man schafft es, unwichtige Dateien auszusortieren. Deshalb kommt jeder Tip recht, welche Dateien man von der Diskette verbannen oder durch gleichwertige kürzere ersetzen könnte.

Unter CP/M Plus gibt es ein Programm, mit dem die verschiedenen nationalen Zeichensätze umgeschaltet werden. Dieses Programm »Language« belegt einigen Platz auf einer Diskette, der sich besser für andere Zwecke nutzen ließe.

Nur leider braucht man das Programm manchmal, so daß es normalerweise nicht einfach von der Diskette gelöscht werden darf. Als Beispiel seien hier Textverarbeitungen genannt, bei denen man mit dem deutschen Zeichensatz arbeitet, unter CP/M jedoch wieder den amerikanischen benötigt. Es gibt allerdings eine direkt einzugebende Steuersequenz, mit der sich die Zeichensätze ebensogut umschalten lassen. Damit wird Language überflüssig.

Um auf den deutschen Zeichensatz umzuschalten (mit Umlauten), war bisher folgende Eingabe notwendig:

```
LANGUAGE 2
```

Bei der neuen Methode gibt man unter CP/M einfach <CTRL+[> gefolgt von einer <2> und der Zahl für den Zeichensatz ein. Die Zahl ist die gleiche wie bei der Umschaltung mit Language. Eine »Submit«-Datei, die vom amerikanischen auf den deutschen Zeichensatz umschaltet, Wordstar aufruft und nach Rückkehr ins CP/M wieder auf den amerikanischen Zeichensatz zurückschaltet, sieht folgendermaßen aus:

```
<CTRL+[> 22 ;deutsche Umlaute
WS          ;Aufruf von Wordstar
<CTRL+[> 20 ;ASCII
```

Mit diesem Trick benötigen Sie das Programm Language nicht mehr und haben den Diskettenplatz für andere Zwecke frei.

Es lassen sich noch weitere Funktionen mit diesem Trick ausführen, wie zum Beispiel einen anderen Bildschirmmodus wählen. Die Befehlssequenzen dafür sind alle im CPC-6128-Handbuch zusammengefaßt. Allgemein lassen sich alle »Escape«-Sequenzen ausführen, in denen keine kleinen Buchstaben vorkommen, denn CP/M wandelt kleine Buchstaben automatisch in Großbuchstaben um.

(Michael Kruse/hf)

Druckeranschluß zweckentfremdet

Am Centronics-Port Ihres CPC läßt sich nicht nur ein Drucker anschließen. Für Bastler ergibt sich hier die Gelegenheit, mit wenig Aufwand Schaltungen anzusteuern.

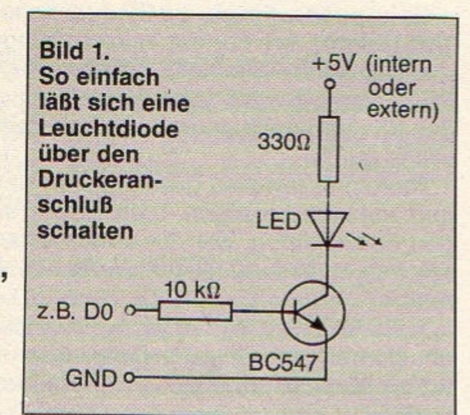


Bild 1. So einfach läßt sich eine Leuchtdiode über den Druckeranschluß schalten

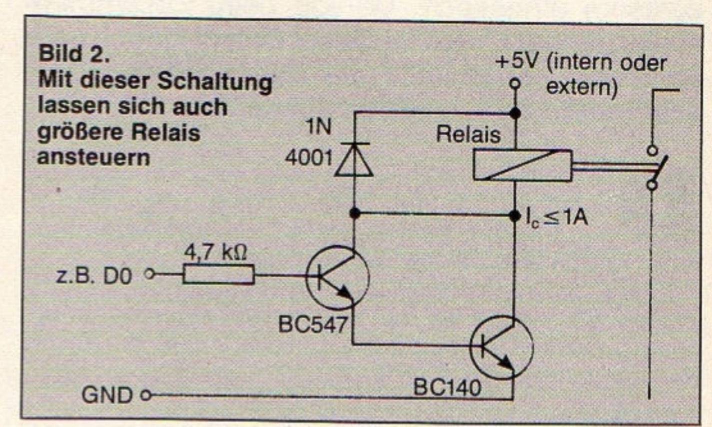


Bild 2. Mit dieser Schaltung lassen sich auch größere Relais ansteuern

Jeder Bastler kennt das Problem: Er möchte eigentlich »nur« eine Kleinigkeit am Computer ausprobieren. Doch weil die Ausgangssignale des CPC am Erweiterungsanschluß nur teilweise gepuffert sind, muß man erst eine Signalpufferung aus Leitungstreibern (Verstärker) und Zwischenspeichern aufbauen, um die Signale des Computers überhaupt verwerten zu können.

Für die Ansteuerung einer Schaltung durch Software ist dann noch die Entwicklung einer Dekodierschaltung erforderlich, die nur auf eine bestimmte Portadresse reagiert. Wie Sie sehen, setzen deshalb selbst kleine Experimente sehr viel Schaltungsaufwand voraus.

Es gibt jedoch eine Lösung, die sowohl hardwaremäßig als auch softwaremäßig wesentlich weniger Aufwand erfordert. Sie müssen dazu nur den Druckeranschluß zweckentfremden.

Der Druckeranschluß ist in der Lage, 8 Bit (die Datenbits D0 bis D6 und das $\overline{\text{STROBE}}$-Signal) parallel auszugeben. Ein Puffern und Zwischenspeichern dieser Signale ist nicht erforderlich, weil dies bereits intern von dem 8-Bit-Register 74LS373 besorgt wird.

Auch die Ansteuerung per Software ist denkbar einfach. Das 8-Bit-Register, das seinen Inhalt auf den Druckeranschluß legt, läßt sich über alle 16-Bit-Adressen, die mit dem Byte EF (hex) beginnen, programmieren.

Der Basic-Befehl

```
OUT &EFFF,&X11111111
```

setzt beispielsweise die Datenbits D0 bis D6 auf 1 und das $\overline{\text{STROBE}}$-Signal auf 0, weil dieses von der internen Elektronik des CPC zusätzlich invertiert wird.

In Maschinensprache läßt sich mit dem OUT-Befehl in der Regel nur eine 8-Bit-Portadresse übergeben. Wenn Sie jedoch den Befehl OUT (C), Register verwenden, wird zu jedem OUT-Befehl gleichzeitig der Inhalt des B-Registers auf die Adreßleitungen A8 bis A15 kopiert, so daß eine 16-Bit-Portadresse entsteht.

So lassen sich beispielsweise mit

```
LD  A,HOFF
LD  B,HOEF
OUT (C),A
```

wie im ersten Beispiel alle Datenbits auf 1 und das $\overline{\text{STROBE}}$-Bit auf 0 setzen, und der Wert 80 (hex) in A legt alle acht Bit auf 0.

Damit Sie auch gleich etwas zum Basteln und Ausprobieren haben, zeigt Bild 1, wie man eine Leuchtdiode über den Druckeranschluß schaltet, und Bild 2, wie sich ein Relais ansteuern läßt.

Die Pinbelegung des Druckeranschlusses finden Sie im Handbuch zum CPC. Für den Anschluß Ihrer Schaltungen empfiehlt es sich, am Druckeranschluß einen Platinenstecker aufzuschieben und an dessen Anschlüsse die Leitungen zu löten, die Sie für Ihre Zwecke benötigen. (ma)

Drucker im Streik

Wenn der Drucker streikt, obwohl das Gerät betriebsbereit ist, der Computer funktioniert und das Druckerkabel richtig steckt, empfiehlt es sich, den DIP-Schalter SLCT IN zu überprüfen.

Stellen Sie sich vor, Ihr Computer sendet Daten an den Drucker, der Drucker ist eingeschaltet und befindet sich in der Betriebsart ON LINE, das Druckerkabel ist ordnungsgemäß befestigt, Druckpapier ist eingespannt – und Ihr Drucker arbeitet trotzdem nicht! In diesem Fall kann es sein, daß der DIP-Schalter, der für das logische Signal SLCT IN zuständig ist, auf der Stellung ON steht. Dadurch läßt sich der Drucker nur mit einem aktiven SELECT-Signal aktivieren. Weil der Schneider CPC dieses Signal jedoch nie erzeugt, ist der Drucker folglich gesperrt.

Ein Umstellen des entsprechenden DIP-Schalters in die Position OFF erweckt den Drucker wieder zum Leben. Zum Umschalten des DIP-Schalters ist es jedoch (für die Druckerelektronik) gesünder, wenn Sie das Gerät vorher ausschalten.

Der angesprochene DIP-Schalter ist übrigens auch eine hilfreiche Maßnahme, um unbefugte Personen von der Benutzung Ihres Druckers abzuhalten. Einfach den DIP-Schalter auf ON stellen, und der kleine Bruder kann nicht mehr testen, wie lange es dauert, bis der Drucker tausend Blatt Endlospapier mit dem Satz »Mein großer Bruder ist doof!« bedruckt hat. (Stefan vom Bruch/ma)

18 KByte mehr

Es ist eine Binsenweisheit, daß man nie genug Speicherplatz auf einer Diskette haben kann. Wir zeigen Ihnen, wie Sie unter CP/M Plus auf allen 3-Zoll-Disketten auch die Diskettenspuren 40 und 41 nutzen können und dadurch pro Seite 9 KByte mehr Speicherplatz erhalten.

Nur ein kleiner Patch für CP/M Plus ist nötig, damit Ihnen im Systemformat 180 KByte und im Data-Only-Format 189 KByte Speicherkapazität pro 3-Zoll-Diskettenseite zur Verfügung stehen. Dieser beachtliche Gewinn an Speicherplatz läßt sich erreichen, indem CP/M Plus nicht – wie sonst üblich – die Spuren 0 bis 39, sondern die Spuren 0 bis 41, also 2 Spuren zusätzlich für die Datenaufzeichnung benutzt.

Dadurch, daß auch Amsdos und CP/M 2.2 die Spuren 40 und 41 anstandslos lesen, steht Ihnen auch unter diesen beiden Betriebssystemen der neue Speicherplatz auf der Diskette zur Verfügung.

Sie müssen zwei kleine Änderungen an CP/M Plus vornehmen, damit die Verarbeitung der zwei neuen Spuren auch dauerhaft in das Betriebssystem eingebunden wird. Im folgenden beschreiben wir detailliert, wie Sie dazu vorgehen müssen, so daß unsere Anleitung auch CP/M-Neulingen keine Schwierigkeiten bereitet. Trotzdem nehmen Sie alle beschriebenen Maßnahmen nur an Dateien vor, von denen Sie auch garantiert Sicherheitskopien besitzen!

1. Starten Sie CP/M Plus.
2. Legen Sie Ihre erste System-Diskette mit der Seite 2 nach oben in das Laufwerk.
3. Laden Sie die Datei »SID.COM« durch Eingabe von »SID« und Drücken der ENTER-Taste.
4. Legen Sie eine Diskette mit der Datei »C10CPM3.EMS« in das Laufwerk.
5. Laden Sie die CP/M-Version mit »#RC10CPM3.EMS« gefolgt von <ENTER>.
6. Geben Sie »#S0DD0« gefolgt von <ENTER> ein und warten Sie, bis »0DD0 AA« erscheint.
7. Tippen Sie nun »B3« und <ENTER> sowie <.> und <ENTER> ein.
8. Darauf geben Sie »S#0DEA« gefolgt von <ENTER> ein und warten, bis »0DEA B3« erscheint.
9. Jetzt tippen Sie »BC« und <ENTER> sowie <.> und <ENTER> ein.
10. Zuletzt müssen Sie die gepatchte CP/M-Version mit »#WC10CPM3.EMS« speichern.

Damit Sie auch die Spuren 40 und 41 beschreiben können, müssen diese erst einmal formatiert werden. Dazu sind zwei kleine Änderungen im Programm »DISCKIT« erforderlich.

1. Legen Sie eine Diskette mit »DISCKIT3.COM« in das Laufwerk.

2. Laden Sie DISCKIT unter SID durch »#RDISCKIT3.COM«.

3. Geben Sie »#S09B3« gefolgt von <ENTER> ein und warten Sie, bis »09B3 27« erscheint.

4. Tippen Sie »29« und <ENTER> sowie <.> und <ENTER> ein.

5. Wählen Sie nun »S#0A13« gefolgt von <ENTER> und warten Sie, bis »0A13 27« erscheint.

6. Jetzt tippen Sie wieder »29« und <ENTER> sowie <.> und <ENTER> ein.

7. Mit »#WDISCKIT3.COM« speichern Sie die neue Version von DISCKIT ab.

Die neue DISCKIT-Version führt fortan die Formatierung bis Spur 41 aus.

Lösen Sie probeweise mit <CTRL+SHIFT+ESC> einen Reset aus, und starten Sie CP/M Plus erneut. Wenn Sie sich jetzt mit dem Befehl SHOW die Laufwerks-Parameter ansehen, muß jede Diskette 9 KByte mehr an Daten aufnehmen können. (Michael Kruse/ma)

Stopp für »Notfälle«

Es gibt im Leben eines Computerbesitzers Situationen, in denen er sich nichts sehnlicher als einen Stopschalter für sein Gerät wünscht, um ein Programm wegen einer Unterbrechung vorübergehend anhalten zu können.

Stellen Sie sich vor, Sie spielen ein Arcade-Spiel und es klingelt an der Tür, oder Sie möchten ein Bildschirmfoto von Ihrem neuen Punkterekord machen, und das Spiel hat keine Pausenfunktion. In beiden Fällen kommt es zu einer mittelschweren Katastrophe, weil Sie den Computer nicht anhalten können.

Es gibt jedoch eine simple Methode, die in diesen Situationen Abhilfe schafft. Sie müssen lediglich am Erweiterungsanschluß des CPC Pin 39 (READY) mit Pin 49 (GND) über einen Schalter vom Typ »ein/aus« verbinden.

Wenn der Schalter offen ist, funktioniert Ihr CPC wie gewöhnlich, doch wenn Sie den Schalter schließen, legt er das READY-Signal, das mit dem WAIT-Eingang des Z80-Prozessors verbunden ist, auf Masse. Dadurch wartet der Prozessor so lange mit der Ausführung des nächsten Befehls, bis das Signal wieder von Masse getrennt beziehungsweise der Schalter geöffnet wird.

Da im Schneider CPC das Auffrischen (Refresh) der dynamischen Speicherbausteine nicht der Prozessor, sondern das Gate Array vornimmt, kann der WAIT-Zyklus beliebig lang ausgedehnt werden, ohne den Inhalt des Arbeitsspeichers zu löschen. Somit hat der Schalter die Funktion eines Stop-Schalters, der ein Programm auf dem CPC jederzeit für einen beliebigen Zeitraum anhalten kann.

Für den Einbau des Stop-Schalters müssen Sie lediglich an Pin 39 und an Pin 49 des Erweiterungsanschlusses einen Draht löten und die beiden Drähte über den Schalter miteinander verbinden. Achten Sie darauf, daß Sie die beiden Drähte so weit vom Platinenrand entfernt auflöten, daß sich auch noch ein Stecker aufschieben läßt. Gehen Sie bitte auch sparsam mit dem Lötzinn um, damit es nicht zu Kurzschlüssen zwischen nebeneinanderliegenden Kontakten kommt. (Boris Lehmann/ma)

Control-Codes unter Kontrolle

Wer ein Listing, das Control-Codes als Steuerzeichen enthält, ausdrucken möchte, bekommt Schwierigkeiten mit dem Druckformat, weil der Drucker die Zeichen zum Teil falsch interpretiert.

Um auch Basic-Programme, die aus Control-Codes bestehende Steuerzeichen enthalten, einwandfrei auszudrucken, ist ein kleiner Trick erforderlich. Das Programm »Ctrllist« durchsucht ein Listing, das zuvor mit »SAVE"@",A« als ASCII-Datei gespeichert wurde, Zeichen für Zeichen nach Control-Codes.

Ist ein Control-Code gefunden, so wird der Buchstabe ermittelt, der zusammen mit der CTRL-Taste gedrückt werden muß, um den Control-Code zu erzeugen. Dieser Buchstabe wird nun anstelle des Control-Codes ausgedruckt.

Damit sich der Buchstabe jedoch von einem herkömmlichen Buchstaben unterscheidet, erscheint er auf dem Ausdruck unterstrichen. Das heißt, beim Eintippen eines auf diese Weise erzeugten Listings ist bei jedem unterstrichenen Buchstaben gleichzeitig die CTRL-Taste zu drücken.

Die im Listing verwendeten Steuerzeichen gelten für Epson-kompatible Drucker wie Star NG-10 und NL-10, DMP-2000 und DMP-3000 etc. Als Besitzer des Schneider-Druckers NLQ 401 müssen Sie statt der Zeichenfolge

"{CTRL+[}R"CHR$(0)

die Kombination

"{CTRL+[}7"

eingeben, um den amerikanischen Zeichensatz auszuwählen.

Drucker ohne Unterstreichungsfunktion können das Programm verarbeiten, indem sie für die Ausgabe der Control-Codes auf Breitschrift geschaltet werden. Für den Seikosha GP-100A gelten beispielsweise folgende Werte:
a$(0)= "{CTRL+O} für Normalschrift und
a$(1)= "{CTRL+N} für Breitschrift.

Anstelle der geschweiften Klammern mit Inhalt, müssen Sie an dieser Stelle im Listing die beiden angegebenen Tasten gleichzeitig drücken. (Dieter Taube/ma)

```
1 WIDTH 255:PRINT#8,"{CTRL+[}R"CHR$(0):a
$(0)="{CTRL+[}-"+CHR$(0):a$(1)="{CTRL+[}
-{CTRL+A}":OPENIN"@":WHILE NOT EOF:LINE
INPUT#9,z$:FOR z=1 TO LEN(z$):x=ASC(MID$
(z$,z)):f=-(x<32):PRINT#8,a$(f)CHR$(x+(6
4+47*(x=31))*f)a$(0);:NEXT:PRINT#8:WEND:
CLOSEIN
```

Listing. Mit diesem Programm gehören Probleme, die Sie bislang mit Control-Codes hatten, der Vergangenheit an

Steckbrief	
Programm:	Ctrllist
Computer:	CPC 464/664/6128
Checksummer:	Explora
Datenträger:	Kassette, Diskette

Bilder mühelos kopiert

Da hat der Anwender mit seinem Malprogramm nun eine Diskette mit Bild-Dateien gefüllt und möchte sie auf eine andere Diskette kopieren. Doch wie funktioniert das Kopieren unter Basic, ohne den Inhalt einer Bild-Datei zu zerstören?

Wer gelegentlich mit einem Malprogramm arbeitet, weiß aus eigener Erfahrung, wie schnell eine Diskette mit selbstgezeichneten Bildern belegt ist. Sobald die erste Arbeitsdiskette voll ist, empfiehlt es sich, für jede Art von Bildern (zum Beispiel Landschaften, Comics, Schaltungen und Konstruktionszeichnungen) eine eigene Diskette anzulegen, damit man später nicht alle Disketten nach einem bestimmten Bild durchsuchen muß.

Doch wie kopiert man Bilder unter Basic von einer Diskette auf eine andere? Im Prinzip funktioniert es ganz einfach, indem man das gewünschte Bild in den Bildschirmspeicher (Adresse C000 bis FFFF hex) des CPC lädt und von dort auf eine andere Diskette schreibt. Doch muß man beachten, daß das zu kopierende Bild nicht durch Betriebssystem-Meldungen oder Abfragen überschrieben wird, weil diese ebenfalls auf den Bildschirm ausgegeben und damit in den Speicherbereich zwischen C000 und FFFF (hex) geschrieben werden.

Unser Listing zeigt den Einzeiler »Bildcopy«, der ein Bild nach Abfrage von Bildschirmmodus und Dateinamen in den Bildschirmspeicher lädt und nach Tastendruck wieder auf eine Diskette schreibt. Während des Kopiervorgangs wird keine Abfrage ausgegeben und die sonst üblichen Kopier-Meldungen lassen sich mit »!« unterdrücken. So wird der Bildschirminhalt und damit das Bild garantiert nicht verändert und kann gefahrlos beliebig oft von einer Diskette auf eine andere kopiert werden.

(Christian Steinbach/ma)

```
1 MODE 2:INPUT"Welcher Bildschirmmodus";
m:IF m<0 OR m>2 THEN RUN ELSE INPUT"Date
iname";n$:IF LEN(n$)<1 OR LEN(n$)>12 THE
N RUN ELSE MODE m:PRINT"Nach dem Laden D
isk wechseln und Taste  druecken!":LOAD"
!"+n$,&C000:CALL &BB06:SAVE"!"+n$,b,&C00
0,&4000:RUN
```

Listing. »Bildcopy« kopiert Bild-Dateien problemlos von einer zu anderen Diskette

Steckbrief	
Programm:	Bildcopy
Computer:	CPC 464/664/6128
Checksummer:	Explora
Datenträger:	Kassette, Diskette

Ladehilfe für Eilige

Ein Einzeiler, der eine Taste im Ziffernblock des Schneider CPC mit einem kleinen Programm belegt, erweist sich als große Hilfe beim Laden von Programmen, die sich auf Diskette befinden.

Wer kennt nicht das Problem: Man möchte ein Programm von Diskette laden, hat jedoch nicht mehr die exakte Bezeichnung der Datei im Kopf. So listet man zunächst das Inhaltsverzeichnis der Diskette mit »CAT«. Falls der Computer jedoch im Modus 1 arbeitet und sich viele Dateien auf der Diskette befinden, ist unter Umständen die Hälfte des Verzeichnisses schon wieder verschwunden, bevor man Zeit hatte, den Inhalt zu studieren.

Deshalb muß der entnervte CPC-Besitzer mit »MODE 2« in den Bildschirmmodus 2 umschalten und das Disketten-Inhaltsverzeichnis erneut listen. Darauf kann er endlich »LOAD"« eingeben und den richtigen Dateinamen eintippen. Sofern er sich nicht verschrieben hat, wird dann auch tatsächlich das gewünschte Programm geladen.

Diese umständliche Prozedur verkürzt der Einzeiler »Catcopy« ganz erheblich. Catcopy ordnet der Punkt-Taste im Ziffernblock ein kleines Programm zu, das sich bei Druck der Taste selbst startet. Das Programm schaltet den Bildschirm in den Modus 2 um, listet das Inhaltsverzeichnis der Diskette im aktuellen Laufwerk und fragt anschließend den Namen des zu ladenden Programms ab. Nun können Sie den COPY-Cursor an den Anfang des gewünschten Dateinamens steuern, die Bezeichnung mit der COPY-Taste übernehmen und durch Drücken der ENTER-Taste den Ladevorgang auslösen.

Die Neubelegung der Punkt-Taste bleibt bis zum Ausschalten des Computers beziehungsweise bis zum nächsten Reset erhalten. Mit dieser Methode lassen sich übrigens auch alle anderen Funktionstasten belegen. Wenn Ihnen weitere hilfreiche Programme einfallen, haben Sie bald eine komplette Utility-Sammlung zusammen, die Sie über die einzelnen Funktionstasten bequem aufrufen können.

(Matthias Kauer/ma)

```
1 KEY 157,"1 ON ERROR GOTO 1:MODE 2:CAT:
INPUT"+CHR$(34)+"Welche Datei"+CHR$(34)+
";a$:IF LEN(a$)>12 THEN RUN ELSE MODE 1:
LOAD a$"+CHR$(13)+"RUN"+CHR$(13):KEY DEF
 7,1,157
```

Listing. Dieses kurze Programm belegt eine Taste des Schneider CPC mit einer Routine zum Laden von Dateien

Steckbrief	
Programm:	Catcopy
Computer:	CPC 464/664/6128
Checksummer:	Explora
Datenträger:	Diskette

Basic-Befehlssatz des CPC auf einen Blick

Für Basic-Programmierer, die es leid sind, stundenlang im Handbuch zu blättern, um einen ganz bestimmten Befehl zu finden, haben wir eine Tabelle zusammengestellt, die die Basic-Befehle des CPC übersichtlich auflistet.

Geht es Ihnen auch so? Sie suchen einen ganz bestimmten Basic-Befehl und wissen auch, daß dieser Befehl im Locomotive-Basic existiert. Doch der Name des Befehls fällt Ihnen partout nicht ein, und die exakte Syntax haben Sie auch nicht mehr im Kopf.

Normalerweise müßten Sie jetzt das CPC-Handbuch hervorkramen und mit der zeitraubenden Suche nach diesem Befehl beginnen. Dank unserer Basic-Befehlstabelle sind diese Zeiten jedoch endgültig vorbei. Die folgende Tabelle listet alle Basic-Befehle des Schneider CPC thematisch gegliedert und innerhalb des Themas alphabetisch geordnet auf. Ebenso sind die Syntax und eine kurze Erklärung zu jedem Befehl angegeben.

Bei den Namen in spitzen Klammern handelt es sich um Parameter (variable Werte). Sind die Parameter zusätzlich in eckige Klammern eingeschlossen, so kann man diese Werte bei Bedarf verwenden. Der zugehörige Text in der Erklärung ist dann ebenfalls in eckigen Klammern eingeschlossen.

Parameter im Detail

Die Namen der verschiedenen Parameter wurden so gewählt, daß sie sich meistens selbst erklären. Ein Verzeichnis am Ende der Tabelle gibt jedoch zusätzlich die exakte Bedeutung zu jedem Parameter mit Wertebereich an.

Ist ein Parameter nicht zwingend erforderlich (optional) und wird er vom Programmierer im Befehl nicht eingesetzt, so nimmt der Basic-Interpreter des CPC für diesen Parameter automatisch den Wert 0 an. In Fällen, wo eine andere Zahl als Standardwert benutzt wird (zum Beispiel bei der Schrittweite einer FOR-NEXT-Schleife), ist dieser Umstand in der Erklärung zu diesem Befehl gesondert vermerkt.

Die Erklärungen zu den einzelnen Befehlen sind aus Platzgründen sehr knapp gehalten. Handelt es sich bei den Befehlen um Funktionen (an der Klammer hinter dem Befehlswort erkennbar), so definiert die Erklärung die Form des Ergebnisses der Funktion. Weil die Sound-Befehle ENT, ENV, SOUND und SQ (<kanal>) sehr komplex sind, wird in der Tabelle aus Platzgründen auf den Grundlagenartikel im 7. Schneider-Sonderheft (ab Seite 150) verwiesen. Die detaillierte Bedeutung der Parameter <format>, <ftaste>, <matrix> und <taste> ist dem Handbuch zu entnehmen. (ma)

Ablaufsteuerung (Schleifen, Sprünge, Verzweigungen etc.)	
Befehl und Parameter	**Bedeutung**
END	Programmende
FOR <variable> = <real1> TO <real2> [STEP <real3>]	Schleife, in der die Laufvariable <variable> ausgehend vom Wert <real1> [mit einer Schrittweite von <real3>] (oder 1) bis auf <real2> herauf- oder heruntergezählt wird
GOSUB <zeile>	Unterprogrammaufruf ab Programmzeile <zeile>
GOTO <zeile>	Aufruf der Programmzeile <zeile>
IF <argument> THEN	<Befehl oder Zeilennummer> [ELSE <Befehl oder Zeilennummer>] Wenn Vergleich oder Ausdruck von <argument> wahr, dann Befehl hinter THEN ausführen bzw. angegebene Zeile aufrufen [sonst Befehl hinter ELSE ausführen bzw. angegebene Zeile aufrufen]
NEXT [<variable>]	Ende der FOR-NEXT-Schleife [mit der Laufvariablen <variable>]
ON BREAK CONT	Blockierung die ESC-Taste um die Unterbrechung eines laufenden Programmes zu verhindern
ON BREAK GOSUB <zeile>	Unterprogrammaufruf ab Programmzeile <zeile>, wenn zweimal die ESC-Taste gedrückt wurde
ON BREAK STOP	Freigabe der ESC-Taste zur Programmunterbrechung (die Wirkung der Befehle ON BREAK CONT und ON BREAK GOSUB wird damit aufgehoben)
ON <argument> GOSUB <zeilen>	Unterprogrammaufruf ab einer Programmzeile von <zeilen>, abhängig von argument (<argument> =1 bewirkt Unterprogrammaufruf ab der ersten aufgeführten Zeilennummer, <argument> =2 bewirkt Aufruf ab der zweiten Zeilennummer usw.)
ON <argument> GOTO <zeilen>	Aufruf einer Programmzeile von <zeilen>, abhängig von argument (<argument> =1 bewirkt Aufruf der ersten aufgeführten Zeilennummer, <argument> =2 bewirkt Aufruf der zweiten Zeilennummer usw.)
RETURN	Ende eines Unterprogramms und Rückkehr in das Hauptprogramm
WEND	Ende der WHILE-WEND-Schleife
WHILE <argument>	Solange der Vergleich von <argument> erfüllt ist, werden die bis zum nächsten WEND folgenden Befehle als Programmschleife ausgeführt

Tabelle. Die Basic-Befehle des Schneider CPC

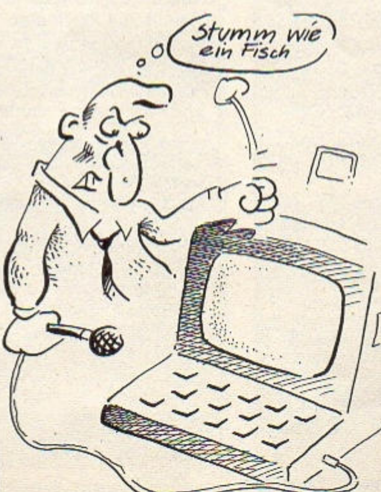
Dateiverwaltung (Diskette, Kassette)	
Befehl und Parameter	**Bedeutung**
CAT	Auflisten des Kassetten- oder Disketteninhaltes
CHAIN <name> [,<zeile>]	Programm <name> wird geladen und [ab Programmzeile <zeile>] ausgeführt
CHAIN MERGE <name> [,<zeile>] [,DELETE <zeile1> - <zeile2>]	Programm <name> wird geladen, in das bestehende Programm eingefügt und [ab Programmzeile <zeile>] ausgeführt. [Die Zeilen <zeile1> bis <zeile2> werden zuvor gelöscht]
CLOSEIN	Geöffnete Eingabedatei schließen
CLOSEOUT	Geöffnete Ausgabedatei schließen
EOF	Zeigt durch den Wert -1 das Dateiende an (sonst auf 0 gesetzt)
LOAD <name> [,<adresse>]	Datei <name> [ab Adresse <adresse>] in den Speicher laden
MERGE <name>	Programm <name> wird geladen und in das bestehende Programm eingefügt
OPENIN <name>	Eingabedatei mit dem Namen <name> öffnen
OPENOUT <name>	Ausgabedatei mit dem Namen <name> öffnen
RUN [<name>]	Programmstart [von Programm <name>]
SAVE <name> [,<typ>] [,<adresse1>,<adresse2>] [,<adresse3>]	Programm <name> [vom Typ <typ>] speichern [Programmbeginn: <adresse1>, Programmlänge: <adresse2>] [und Startadresse: <adresse3>]
SPEED WRITE <boolean>	Schaltet Schreibgeschwindigkeit für Kassettenrecorder um

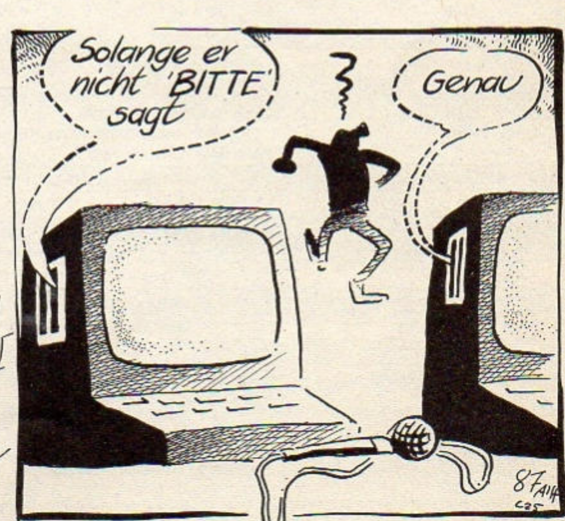
Ein und Ausgabe (Bildschirm, Drucker, Speicher, Ports)

Befehl und Parameter	Bedeutung
CLEAR INPUT	Vorherige Eingaben löschen
CLS[# <gerät>]	Einheit [<gerät>] löschen
COPYCHR$(# <gerät>)	Zeichen auf aktueller Position in Einheit <gerät>
CURSOR [<boolean1>] [,<boolean2>]	Systemcursor [abhängig von <boolean1>] und Benutzercursor [abhängig von <boolean2>] ein- oder ausschalten
INKEY(<taste>)	Fragt ab, ob die Taste mit der Nummer <taste> nicht gedrückt (-1), gedrückt (0), zusammen mit SHIFT gedrückt (32), zusammen mit CTRL gedrückt (128) oder zusammen mit SHIFT und CTRL gedrückt (160) wird
INKEY$	Text der momentan gedrückten Taste
INP(<adresse>)	Auf Eingabeport <adresse> anliegender Wert
INPUT[# <gerät>,] [<text>;] <variablen>	[Ausgabe von <text> und] Abfrage der Werte für die Variablen <variablen> [über das Gerät <gerät>]
LINE INPUT[# <gerät>,] [<text>;] <variable>	[Ausgabe von Text <text> und] Abfrage einer Textzeile für die Variable <variable> [über das Gerät <gerät>]
LOCATE[# <gerät>,] <x>,<y>	Positioniert den Textcursor auf dem Punkt <x>,<y> [von Fenster <gerät>]
MODE <mode>	Auswahl von Bildschirmmodus <mode>
OUT <adresse>,<shortint>	Ausgabe des Wertes <shortint> über Ausgabeport <adresse>
PEEK (<adresse>)	Inhalt der Speicherzelle <adresse> des Arbeitsspeichers
POKE <adresse>,<shortint>	Wert von <shortint> in Speicherzelle <adresse> des Arbeitsspeichers schreiben
POS# <gerät>	Aktuelle Position des Cursors, Druckkopfes oder des Dateizeigers auf Gerät <gerät>
PRINT[# <gerät>,] <zeichen>	Ausdruck von Text und/oder Variablen [über Gerät <gerät>]
PRINT[# <gerät>,] [<zeichen>] SPC <shortint> <zeichen>	Ausdruck von [Text und/oder Variablen] [über Gerät <gerät>,] <shortint> Leerzeichen und nachfolgendem Text und/oder Variablen
PRINT[# <gerät>,] [<zeichen>] TAB <shortint> <zeichen>	Ausdruck von [Text und/oder Variablen] [über Gerät <gerät>,] Text und/oder Variablen ab Zeilenposition <shortint>
PRINT USING <format>; <real>	Ausdruck des Wertes von <real> im Format <format>
VPOS(# <gerät>)	Vertikale Position des Textcursors in Fenster <gerät>
WAIT <adresse>,<shortint1> [,<shortint2>]	Wartet auf die Dateneingabe über Port <adresse> [,verknüpft das eingelesene Byte eventuell mit <shortint2> über die XOR-Funktion] und verknüpft das Ergebnis mit <shortint1> über die AND-Funktion
WIDTH <shortint>	Anzahl der Zeichen pro Zeile wird beim Drucken auf <shortint> begrenzt
WINDOW[# <gerät>,] <xL>, <xR>,<yO>,<yU>	Definiert Fenster [mit der Nummer <gerät>] in den Grenzen <xL> (linke Spalte), <xR> (rechte Spalte), <yO> (obere Zeile) und <yU> (untere Zeile)
WINDOW SWAP <gerät1>, <gerät2>	Fenster <gerät1> wird gegen Fenster <gerät2> ausgetauscht
WRITE[# <gerät>,] <zeichen>	Ausdruck von durch Kommata getrennten Texten und/oder Variablen [über Gerät <gerät>]
ZONE <shortint>	Spaltenbreite auf dem Bildschirm wird auf <shortint> Zeichen festgesetzt (die Spalteneinteilung steuert das Komma im PRINT-Befehl)

Farbe und Grafik

Befehl und Parameter	Bedeutung
BORDER <farbe1> [,<farbe2>]	Rand in Farbe <farbe1> [die periodisch mit Farbe <farbe2> wechselt]
CLG [<register>]	Grafikbildschirm [mit Farbe aus INK-Register <register>] füllen
DRAW <integer1>, <integer2> [,<register>] [,<modus>]	Linie von der aktuellen Position des Grafikcursors bis zum Koordinatenpunkt <integer1>,<integer2> [in der Farbe des INK-Registers <register>][mit der Verknüpfung <modus>] ziehen
DRAWR <integer1>, <integer2> [,<register>] [,<modus>]	Linie von der aktuellen Position des Grafikcursors zum relativen Koordinatenpunkt <integer1>,<integer2> [in der Farbe des INK-Registers <register>] [mit der Verknüpfung <modus>] ziehen
FILL <register>	Füllt Fläche, in der die aktuelle Position des Grafikcursors liegt, mit der Farbe aus INK-Register <register>
FRAME	Synchronisation von Grafikaufbau und Strahlrücklauf
GRAPHICS PAPER <register>	Hintergrund der Grafik in Farbe des INK-Registers <register> darstellen
GRAPHICS PEN <register> [,<boolean>]	Vordergrund der Grafik in Farbe des INK-Registers <register> darstellen
INK <register>,<farbe1> [,<farbe2>]	INK-Register <register> wird mit der Farbe <farbe1> [und der Farbe <farbe2>] geladen
MASK <shortint> [,<boolean>]	Die Bits des Wertes von <shortint> definieren die Schablone für das Zeichnen von Grafiklinien
MOVE <integer1>, <integer2> [,<register>] [,<modus>]	Unsichtbare Verschiebung des Grafikcursors von der aktuellen Position zum Koordinatenpunkt <integer1>,<integer2> [mit Wahl der Farbe aus INK-Register <register>] [und der Verknüpfung <modus>]
MOVER <integer1>, <integer2> [,<register>] [,<modus>]	Unsichtbare Verschiebung des Grafikcursors von der aktuellen Position auf den relativen Koordinatenpunkt <integer1>, <integer2> [mit Wahl der Farbe aus INK-Register <register>] [und der Verknüpfung <modus>]
ORIGIN <integer1>, <integer2> [,<integerL>, <integerR>,<integerO>, <integerU>]	Festlegen des Koordinaten-Nullpunktes auf der absoluten Bildschirm-Koordinate <integer1>,<integer2> (0,0=links unten) [Festlegen des Grafikfensters mit dem rechten Rand auf Position <integerR>, dem linken Rand auf Position <integerL>, dem oberen Rand bei <integerO> und dem unteren Rand bei <integerU>]





Farbe und Grafik

Befehl und Parameter	Bedeutung
PAPER[# <gerät>,] <register>	Legt die Hintergrundfarbe [für Fenster <gerät>] auf die Farbe aus INK-Register <register> fest
PEN[# <gerät>,] <register> [,<boolean>]	Legt die Zeichenfarbe [für Fenster <gerät>] auf die Farbe aus INK-Register <register> fest [und schaltet den Transparentmodus abhängig von <boolean> ein oder aus]
PLOT <integer1>,<integer2> [,<register>] [,<modus>]	Grafikpunkt auf die Koordinatenposition <integer1>,<integer2> [in der Farbe des INK-Registers <register>] [mit der Verknüpfung <modus>] setzen
PLOTR <integer1>, <integer2> [,<register>] [,<modus>]	Grafikpunkt auf die relative Koordinatenposition <integer1>,<integer2> [in der Farbe des INK-Registers <register>] [mit der Verknüpfung <modus>] setzen
SPEED INK <shortint1>, <shortint2>	Die erste Farbe im aktuellen INK-Register ist jeweils 0,02 x <shortint1> Sekunden und die zweite Farbe jeweils 0,02 x <shortint2> Sekunden
TAG[# <gerät>]	Textausgabe erfolgt auf die Position des Grafikcursors [in Fenster <gerät>]
TAGOFF[# <gerät>]	Textausgabe erfolgt auf die Position des Textcursors [in Fenster <gerät>]
TEST(<integer1>, <integer2>)	Farbnummer des Punktes auf der Grafikposition <integer1>,<integer2>
TESTR(<integer1>, <integer2>)	Farbnummer des Punktes auf der relativen Grafikposition <integer1>,<integer2>
XPOS	Horizontale Ordinate des Grafikcursors
YPOS	Vertikale Ordinate des Grafikcursors

Fehlerbehandlung

Befehl und Parameter	Bedeutung
DERR	Fehlercode des letzten Diskettenfehlers
ERL	Nummer der Zeile, in der ein Fehler aufgetreten ist
ERR	Nummer des zuletzt aufgetretenen Fehlers
ERROR <shortint>	setzt ERR auf den Wert <shortint>
ON ERROR GOTO <zeile>	Aufruf der Programmzeile <zeile>, wenn ein Fehler während des Programmlaufs auftritt
RESUME [<zeile>]	Programm wird nach einem ON ERROR GOTO-Befehl in der fehlerverursachenden Zeile [ab der Programmzeile <zeile>] fortgesetzt
RESUME NEXT	Programm wird nach einem ON ERROR GOTO-Befehl fortgesetzt, als wäre kein Fehler aufgetreten

Zahlenverarbeitung (Logische Befehle, mathematische Funktionen etc.)

Befehl und Parameter	Bedeutung
ABS(<real>)	Absolutwert von <real>
<argument1> AND <argument2>	Logische AND-Verknüpfung von <argument1> mit <argument2>
ATN(<real>)	Arcustangens im Bogenmaß von <real>
COS(<real>)	Cosinus von <real>
DEC$(<real>,<format>)	Wert von <real> im Format <format>
DEF FN<name> [(<variable>)]= <argument>	Definition der Funktion <name> mit der Anweisung <argument>
DEG	Umschalten auf Winkelgradmaß
EXP(<real>)	Basis der natürlichen Logarithmen (e) potenziert mit <real>
FIX(<real>)	Wert vor dem Komma von <real>
FN<name>[(<variablen>)]	Ergebnis der Funktion <name>
INT(<real>)	Rundet <real> auf den ganzzahligen Wert ab
LOG(<real>)	Natürlicher Logarithmus von <real>
LOG10(<real>)	Dekadischer Logarithmus von <real>
MAX(<numeri>)	Höchster Wert der Zahlen von <numeri>
MIN(<numeri>)	Kleinster Wert von <numeri>
<argument1> MOD <argument2>	Rest der Division von <argument1> mit <argument2>
NOT <argument>	Invertierung des logischen Wertes von <argument>
<argument1> OR <argument2>	Logische OR-Verknüpfung von <argument1> mit <argument2>
PI	dezimaler Wert der Kreiszahl

Befehl und Parameter	Bedeutung
RAD	Umschalten auf Bogenmaß
RANDOMIZE <real>	Anfangswert für Zufallszahlengenerator
RND(<real>)	Zufallszahl (Wiederholung der letzten Zahl, wenn <real>=0)
ROUND(<real>,<bereich>)	Der Wert von <real> wird abhängig von <bereich> auf die Anzahl von <bereich> Nachkommastellen (<bereich> negativ) oder auf <bereich> Vorkommastellen (<bereich> positiv) gerundet
SGN(<real>)	1, wenn <real> größer als 0; -1, wenn <real> kleiner als 0; 0, wenn <real> gleich 0
SIN(<real>)	Sinus von <real>
SQR(<real>)	Wurzel von <real>
TAN(<real>)	Tangens von <real>
UNT(<adresse>)	Wert von <adresse> in den üblichen Wertebereich von <integer> (-32768 bis +32767) umgewandelt
<argument1> XOR <argument2>	Logische XOR-Verknüpfung (Exklusiv-Oder) von <argument1> mit <argument2>

Programmierung

Befehl und Parameter	Bedeutung
AUTO <zeile1> [,<zeile2>]	Automatische Erzeugung von Zeilennummern ab <zeile1> [mit Schrittweite <zeile2>]
CONT	unterbrochenes Programm fortsetzen
DELETE <zeile1> [- <zeile2>]	Programmzeile <zeile1> [bis <zeile2>] löschen
EDIT <zeile>	Programmzeile <zeile> zum Editieren aufrufen
HIMEM	Höchste von Basic belegte Speicheradresse
LIST [<zeile1>] [-<zeile2>] [,# <gerät>]	Listet Programm [von Zeile <zeile1>] [bis <zeile2>] [auf das Gerät <gerät>]
MEMORY <adresse>	Höchste freie Speicheradresse für Basic-Programme und Basic-Variablen gleich <adresse>
NEW	Löschen des im Arbeitsspeicher befindlichen Programmes
REM	Der dem REM-Befehl nachfolgende Text dient zur Programmdokumentation und wird vom Interpreter nicht berücksichtigt
RENUM [<zeile1>] [,<zeile2>] [,<zeile3>]	Neunumerierung des Programmes [ausgehend von der neuen Zeilennummer <zeile1>] (oder 10) [ab der alten Programmzeile <zeile2>] [mit der Schrittweite <zeile3>] (oder 10)
RUN [<zeile>]	Programmstart [ab Programmzeile <zeile>]
STOP	Programm wird unterbrochen und kann mit CONT fortgeführt werden
TROFF	Trace-Funktion ausgeschaltet
TRON	Trace-Funktion eingeschaltet (während eines Programmlaufs wird die Zeilennummer jeder bearbeiteten Programmzeile auf den Bildschirm ausgegeben)

Stringverarbeitung

Befehl und Parameter	Bedeutung
INSTR([<shortint>,] <string1>,<string2>)	Position, an der <string2> in <string1> enthalten ist [Suche ab Position <shortint> in <string1>]
LEFT$(<text>,<shortint>)	Die ersten <shortint> Zeichen von Text <text>
LEN(<text>)	Anzahl der Zeichen in Text <text>
LOWER$(<text>)	Text in Kleinbuchstaben von <text>
MID$(<text>,<shortint1> [<shortint2>])	Zeichenkette aus <text> ab der Position <shortint1> [mit der Länge <shortint2>] (oder bis zum Textende)
MID$(<variable>, <shortint1>, <shortint2>)= <text>	Fügt <shortint2> Zeichen des Textes <text> ab der Position <shortint1> in die Stringvariable <variable> ein
RIGHT$(<text>,<shortint>)	Die letzten <shortint> Zeichen von Text <text>
SPACE$(<shortint>)	String bestehend aus <shortint> Leerzeichen
STR$(<real>)	Wert von <real> in ein String umgewandelt
STRING$(<shortint>, <text>)	Vervielfachung des ersten Zeichens von <text> um den Faktor <shortint>
UPPER$(<text>)	Text in Großbuchstaben von <text>

Töne und Rauschen

Befehl und Parameter	Bedeutung
ENT	Frequenzhüllkurve (siehe 7. Schneider-Sonderheft, Seite 150)
ENV	Lautstärkenhüllkurve (siehe 7. Schneider-Sonderheft, Seite 150)
ON SQ (<kanal>) GOSUB <zeile>	Unterprogrammaufruf ab Programmzeile <zeile>, wenn in der Tonwarteschlange von Kanal <kanal> mindestens ein Platz frei ist
RELEASE <kanal>	Wartezustand von Kanal <kanal> aufheben
SOUND <parameter>	Definition von Ton und Rauschen (siehe 7. Schneider-Sonderheft, Seite 150)
SQ(<kanal>)	Zustand von Kanal <kanal> (siehe 7. Schneider-Sonderheft, Seite 150)

Unterbrechungen (Interrupt und Uhren)

Befehl und Parameter	Bedeutung
AFTER <integer> [,<clock>] GOSUB <zeile>	Wenn auf der Uhr [<clock>] 0,02 x <integer> Sekunden verstrichen sind, erfolgt ein Unterprogrammaufruf ab Zeile <zeile>
DI	Unterbrechungen durch AFTER und EVERY sperren
EI	Unterbrechungen durch AFTER und EVERY freigeben
EVERY <integer> [,<clock>] GOSUB <zeile>	Immer, wenn auf der Uhr [<clock>] 0,02 x <integer> Sekunden abgelaufen sind, erfolgt ein Unterprogrammaufruf ab Zeile <zeile>
REMAIN(<clock>)	Restzeit von Uhr <clock> (Uhr wird zusätzlich abgeschaltet)
TIME	Zeit, die seit Einschalten des Computers verstrichen ist in der Einheit 1/300 Sekunde

Variablenverarbeitung (Definition, Umrechnung etc.)

Befehl und Parameter	Bedeutung
ASC(<string>)	Numerischer Wert des ersten Zeichens von <string>
BIN$(<integer> [,<integer>])	Binärer Wert von <integer> [mit <stellen> Ziffern]
CHR$(<shortint>)	ASCII-Zeichen von <shortint>
CINT(<shortreal>)	Gerundeter Integer-Wert von <shortreal>
CLEAR	Alle Variablen auf 0 setzen und alle Dateien vergessen
CREAL(<real>)	Real-Wert von <real>
DEFINT <variablen>	Definition von Interger-Variablen
DEFREAL <variablen>	Definition von Real-Variablen
DEFSTR <variablen>	Definition von String-Variablen
DIM <variable>(<index>)	Definition des Datenfeldes <variable>
ERASE <variablen>	Variablen <variablen> löschen
FRE(0)	Größe des von Basic nicht benutzten Speicherplatzes
FRE(" ")	Größe des von Basic nicht benutzten Speicherplatzes mit Auslösung einer Garbage Collection
HEX$(<integer> [,<stellen>])	Hexadezimaler Wert von <integer> [mit <stellen> Ziffern]
LET <variable> = <argument>	Der Variablen <variable> wird das Ergebnis von <argument> zugewiesen
VAL(<string>)	Numerischer Wert von <string> in eine Zahl umgewandelt

Sonstige Befehle

Befehl und Parameter	Bedeutung
CALL <integer> [,<parameter>]	Maschinencode-Aufruf ab Adresse <integer> [und Übergabe der Werte <parameter>]
DATA <konstanten>	Datenfeld bestehend aus den Daten <konstanten>
JOY(<boolean>)	Auf Joystick <boolean> wurde »hoch« (1), »runter« (2), »links« (4), »rechts« (8), »feuer1« (16), »feuer2« (32) oder eine Kombination (Summe der Werte) gedrückt
KEY <ftaste>,<text>	Belegt Funktionstaste <ftaste> mit dem Text <text>
KEY DEF <taste>, <boolean> [,<taste1>] [,<taste2>] [,<taste3>]	Die Taste mit der Nummer <taste> erzeugt den Text von Taste <taste1>, zusammen mit SHIFT den Text von <taste2> und zusammen mit CTRL den Text von <taste3>
READ <variablen>	Werte aus DATA-Zeilen einlesen und den Variablen <variablen> zuweisen

Befehl und Parameter	Bedeutung
RESTORE [<zeile>]	Zurücksetzen des DATA-Zeigers [auf den ersten Wert der Programmzeile <zeile>]
SPEED KEY <shortint1>, <shortint2>	Repeat-Funktion startet nach 0,02 x <shortint1> Sekunden und erzeugt alle 0,02 x <shortint2> Sekunden ein Zeichen
SYMBOL <shortint>, <matrix>	Definition des Zeichens mit dem ASCII-Code <shortint> als Bildpunktmuster der Werte von <matrix>
SYMBOL AFTER <shortint>	Zeichen ab dem ASCII-Code <shortint> können über SYMBOL frei definiert werden

Anmerkung zu den Grafikbefehlen:
Ein relativer Koordinatenpunkt bezieht sich auf die aktuelle Position des Grafikcursors als Koordinaten-Nullpunkt

Befehl und Parameter	Bedeutung
<adresse>	Speicher- oder Portadresse (0 bis 65535)
<argument>	Mathematischer Ausdruck oder logischer Vergleich (1=wahr, 0=falsch)
<bereich>	Stellenbereich, innerhalb dessen ein Wert gerundet wird
<boolean>	<boolean>=0: Aus (CURSOR), nicht transparent (GRAPHICS PEN und PEN), Joystick 0 (JOY), nicht wiederholen (KEY DEF), ersten Punkt nicht setzen oder 1000 Baud (SPEED WRITE) <boolean>=1: Ein (CURSOR), transparent (GRAPHICS PEN und PEN), Joystick 1 (JOY), wiederholen (KEY DEF), ersten Punkt setzen (MASK) oder 2000 Baud (SPEED WRITE)
<clock>	Nummer der internen Uhren des CPC (0 bis 3)
<farbe>	Farbnummer (0 bis 26)
<format>	Spezieller Textausdruck, der das Format einer Datenausgabe regelt (Näheres siehe Handbuch)
<ftaste>	Nummer einer Funktionstaste (128 bis 159, siehe Handbuch)
<gerät>	Ein-/Ausgabegerät wie Fenster, Drucker oder Kassettenrecorder (0 bis 9)
<index>	Ganzzahliger Wert oder ganzzahlige Werte, die die Anzahl der Felder eines Datenfeldes bestimmen
<integer>	Ganzzahliger Wert (-32768 bis +32767, manchmal bis +65535)
<kanal>	1=Kanal A, 2=Kanal B, 4=Kanal C
<konstanten>	Feste Integer-, Real- oder Stringwerte
<matrix>	8 Werte vom Typ <shortint>, die das Bildpunktmuster eines Text- oder Grafikzeichens definieren (näheres siehe Handbuch)
<mode>	Ganzzahliger Wert, der den Bildschirmmodus (0, 1 oder 2) bestimmt
<modus>	Art der logischen Verknüpfung (1=XOR, 2=AND, 3=OR)
<name>	Zulässiger Datei- oder Funktionsname
<numeri>	Feste Integer- oder Real-Werte
<parameter>	bis zu 32 Integer-Werte
<real>	Reele Zahl (-1,7 x 10^{38} bis +1,7 x 10^{38})
<register>	INK-Registernummer (0 bis 15)
<shortint>	Ganzzahliger Wert (0 bis 255)
<shortreal>	Reele Zahl (-32768 bis +32767)
<stellen>	Anzahl der Ziffern einer Zahl (1 bis 16)
<string>	Alphanumerische Zeichenkette
<taste>	Tastennummer (0 bis 79, siehe Handbuch)
<text>	Von Anführungszeichen eingeschlossene alphanumerische Zeichenkette oder String-variable
<typ>	A=ASCII-Datei, B=Binärdatei, P=geschütztes Basic-Programm
<variable>	Name einer Variablen
<variablen>	Name einer oder mehrerer durch Komma getrennten Variablen
<x>	X-Koordinate (1 bis 80)
<y>	Y-Koordinate (1 bis 25)
<zeichen>	Von Anführungszeichen eingeschlossene alphanumerische Zeichenkette und/oder Variablen
<zeile>	Zeilennummer einer Basic-Programmzeile (1 bis 65535)
<zeilen>	Liste von Zeilennummern für Basic-Programmzeilen

Tabelle. Die Basic-Befehle des Schneider CPC (Schluß)

1000 Berlin

DATAPLAY
Bundesallee 25 · 1000 Berlin 31
Telefon (030) 8826322
ATARI Commodore Schneider NEC

Digital-Computer
Knesebeckstr. 76 · 1000 Berlin 12
Telefon
030-8827791

2000 Hamburg

2210 Itzehoe

Der Computerladen
Inhaber Ulrich Bübel Martin Kopplow
Coriansberg 2 · 2210 Itzehoe
Telefon (04821) 3390/91

3000 Hannover

trend DATA Computer
IBM · EPSON · TRIUMPH ADLER
HEWLETT PACKARD · ATARI etc.
trendDATA Computer GmbH
Am Marstall 18–22 · 3000 Hannover 1
Telefon (05 11) 1 66 05-0

3502 Vellmar

4000 Düsseldorf

Joysoft
Humboldtstr. 84
4000 Düsseldorf
☎ 0211/6801403

4322 Sprockhövel

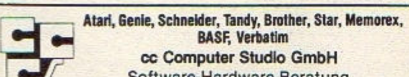
Schneider COMPUTER DIVISION
vortex
RALF HILLE DATENTECHNIK
Mittelstraße 61, 4322 Sprockhövel 2, Telefon (02339) 7191

4400 Münster

4430 Steinfurt

ATARI SCHNEIDER STAR EPSON
Computer
Büromaschinen
Service
4430 Steinfurt · Tecklenburger Str. 27
Telefon: 02551/2555

4600 Dortmund



cc Computer Studio GmbH
Software-Hardware-Beratung
Service-Eilversand
Ihre Ansprechpartner: v. Schablinski, Jan P. Schneider
Elisabethstraße 5
4600 Dortmund 1
T. 0231/528184 · Tx 822631 cccsd

4972 Löhne

5000 Köln

Joysoft
Berrenratherstr. 159
5000 Köln 41
☎ 02 21 / 41 66 34

Große Umfrage

Hat Ihnen das Heft gefallen?

Wieder einmal haben Sie ein Schneider-Sonderheft von Happy-Computer vor sich liegen. Und wieder fragen wir uns, ob wir mit unseren Themen richtig liegen. Denn diese Frage können nur Sie - unser Leser - beantworten. Deshalb schicken Sie uns bitte den untenstehenden Fragebogen ausgefüllt zurück. Denn seine Auswertung zeigt uns den Weg, den wir mit dem nächsten Schneider-Sonderheft einschlagen müssen.

Auch der Schneider-Teil im Stamm-Magazin Happy-Computer wird nach Ihren Vorschlägen gestaltet. Deshalb ist Ihre Meinung für uns so immens wichtig.

Auch Sie profitieren also davon, wenn Sie uns Ihre - positive und negative - Kritik wissen lassen.

Schicken Sie den ausgefüllten Fragebogen bitte an:

Markt & Technik Verlag AG
Redaktion Happy-Computer
Kennwort: Schneider-Umfrage
Hans-Pinsel-Straße 2
8013 Haar bei München

(ja)

Fragebogen zum 8. Schneider-Sonderheft

Wie hat Ihnen dieses Heft gefallen?

- ☐ sehr gut
- ☐ gut
- ☐ mittel
- ☐ weniger gut
- ☐ gar nicht

Welche Rubriken wollen Sie in Zukunft erweitert sehen?

- ☐ Hardware
- ☐ Software
- ☐ Basteleien
- ☐ Spiele-Tests
- ☐ CP/M
- ☐ Grundlagen
- ☐ Einsteiger-Teil
- ☐ Aktuelles
- ☐ Tips&Tricks
- ☐ Spiele-Listings
- ☐ Anwendungs-Listings
- ☐ Grafik-Listings

Welche Rubriken sollen in Zukunft eingeführt werden?

____

____

Welche Computer-Zeitschriften lesen Sie?

- ☐ Happy-Computer
- ☐ deutsche Schneider-Zeitschriften - wenn ja, welche?

____

____

- ☐ englische Amstrad-Zeitschriften
- ☐ andere - wenn ja, welche?

____

____

Welche Schneider-Sonderausgaben von Happy-Computer haben Sie sich schon gekauft?

- ☐ 1. Schneider-Sonderheft
- ☐ 2. Schneider-Sonderheft
- ☐ 3. Schneider-Sonderheft
- ☐ 4. Schneider-Sonderheft
- ☐ 5. Schneider-Sonderheft
- ☐ 6. Schneider-Sonderheft
- ☐ 7. Schneider-Sonderheft
- ☐ 8. Schneider-Sonderheft

Wenn Sie alle acht Schneider-Sonderhefte besitzen, welches hat Ihnen am besten gefallen?

- ☐ 1. Schneider-Sonderheft
- ☐ 2. Schneider-Sonderheft
- ☐ 3. Schneider-Sonderheft
- ☐ 4. Schneider-Sonderheft
- ☐ 5. Schneider-Sonderheft
- ☐ 6. Schneider-Sonderheft
- ☐ 7. Schneider-Sonderheft
- ☐ 8. Schneider-Sonderheft

Welchen Computer besitzen Sie?

- ☐ Schneider CPC 464
- ☐ Schneider CPC 664
- ☐ Schneider CPC 6128
- ☐ Schneider Joyce
- ☐ Schneider PC
- ☐ einen anderen, welchen?

____

Welchen Diskettencontroller besitzen Sie?

- ☐ Schneider
- ☐ Vortex
- ☐ Vortex X-Controller
- ☐ einen anderen, welchen?

____

Welche Speichererweiterung besitzen Sie?

- ☐ Data Media
- ☐ dk'tronics
- ☐ Vortex
- ☐ eine andere, welche?

____

Ich bin damit einverstanden, daß die hier gemachten Angaben elektronisch verarbeitet werden.

Name/Vorname ____________________

Straße ____________________

PLZ/Ort ____________________

Alter ____ Jahre

Giro-Konto in Unordnung

Aus vielen Anrufen voller Lob, Kritik und Fragen zum Programm »Giro« aus dem 6. Schneider-Sonderheft (Happy-Computer-Sonderausgabe 13, Seite 75) kristallisierte sich ein kleiner Fehler heraus, den wir natürlich sehr bedauern und hier ausmerzen.

Fügen Sie in das Listing die neue Zeile

```
1835 e2=e(k)+d(k):IF e2
<12 THEN e2=e2-12
```

ein und ändern Sie den Anfang der Zeile 1840 wie folgt

```
1840 IF e(k)=n OR e2=n
THEN ...
```

Ohne diese Änderung erscheint eine zweimonatliche Fixbuchung im ersten Monat der Vorschau, nicht jedoch im dritten. Nach der Neuanlage einer Konto-Datei mußten Sie bislang auch die Statistik extra neu starten, damit das Datum in der Kopfzeile der Statistik zur Anzeige gelangt. Die folgende Änderung macht diesen zusätzlichen Schritt unnötig

```
430 ...a1$=a$:b4$=b3$:
flag=1
```

Bei der Handhabung des Programms sind noch einige Punkte zu beachten, um Fehlerquellen zu vermeiden. So müssen Sie sich bei Neuanlage einer Datei die Fixbuchungen im Vormonat der ersten Gehaltsbuchung anzeigen lassen, also zum Beispiel:

```
31.07.87 Anlegen der
         Fixbuchungen
01.08.87 Gehaltseingabe
```

Zur Datumseingabe ist das Format »TTMMJJ« zwingend vorgeschrieben (je eine zweistellige Zahl für Tag, Monat und Jahr ohne Trennung direkt hintereinander). Da aber Giro die Einhaltung dieses Formats nicht prüft, führen falsche Eingaben (beispielsweise mit Punkten: »31.12.87«) zu falschen Berechnungen.

Daneben ist zu beachten, daß die Gehaltseingabe Vorrang gegenüber jeder anderen Buchung hat.

(Ingo Strecker/ja)

Keine Backups

Daß uns ein bedauerlicher Fehler bei der Beschreibung des Listings »Backup-Master« im 6. Schneider-Sonderheft unterlief, haben die davon betroffenen Leser längst bemerkt. Der dortige Hinweis auf die Lauffähigkeit des Programms auf den CPCs 664 und 6128 war schlicht fehl am Platz, denn aufgrund direkter Aufrufe von CPC 464-ROM-Routinen arbeitet der Backup-Master natürlich nicht auf diesen Computern. Weil sich jedoch viele Besitzer dieser CPC-Typen nach Lektüre der vielversprechenden Anleitung an das Abtippen machten, den Mangel entdeckten und uns, teils verzweifelt, teils entrüstet mit Briefen und Anrufen zudeckten, fühlten wir uns in die Pflicht genommen, schnellstmöglich für Abhilfe zu sorgen. Bereits im letzten Schneider-Sonderheft, spätestens jedoch an dieser Stelle sollten Sie den entsprechenden Nachhall finden. Leider gestaltete sich die Anpassung aber nicht nur aufwendiger, sondern auch schwieriger als anfangs abzusehen war, so daß sie uns zum Redaktionsschluß dieser Ausgabe noch nicht vorlag. Interessierte Leser können sie bei uns schriftlich anfordern. Senden Sie uns dazu bitte eine genaue Beschreibung Ihres Computersystems: Computertyp, Diskettenlaufwerk und eventuelle Erweiterungen. Adressieren Sie die Sendung an den Verlag, Redaktion Happy-Computer, zu Händen Thomas Jacobi. Wir schicken Ihnen dann das Listing des passenden Patches. Noch einfacher wird die Sache für Sie, wenn Sie Ihrer Sendung eine formatierte 3-Zoll- oder Vortex-5-Zoll-Diskette beilegen.

(ja)

Scale für alle

Für das Programm »Scale« aus dem 6. Schneider-Sonderheft, Seite 92 erreichten uns inzwischen mehrere Anpassungen für den CPC 6128. Die Version von Wolfgang Ried aus Feucht wollen wir Ihnen nicht vorenthalten.

Die nötigen Änderungen beschränken sich im wesentlichen auf zwei Problemkreise. Das sind die Anpassung differierender Adressen von Betriessystemvariablen und der Systemroutine zur vorzeichenlosen Multiplikation. Letztere lagerten die Entwickler der

Inserenten in dieser Ausgabe:

Bitte beachten Sie auch unseren Einkaufsführer auf den Seiten 156/157.

beiden neueren CPC-Modelle ins Basic-ROM aus. Um Scale nun auch auf diesen Computern zugänglich zu machen, müssen Sie zwei neue Zeilen hinzufügen:

```
102 DATA 9C30,CD,00,B9,CD,
    72,DD,CD,03,7C0D
103 DATA 9C38,B9,C9,00,00,
    00,00,00,00,6EC0
```

Sie enthalten eine Routine, die anstelle der vorzeichenlosen Multiplikation aufgerufen wird und das ROM ein- beziehungsweise wieder ausschaltet. Die restlichen Änderungen betreffen mehrere Zeilen, deren Inhalt Sie im folgenden finden.

```
108 DATA 9C60,16,C3,93,CA,
    F3,3A,2F,B7,2299
109 DATA 9C68,32,A3,B6,3A,
    30,B7,32,A4,273C
110 DATA 9C70,B6,CD,C6,BB,
    ED,53,AC,AC,7CE0
111 DATA 9C78,22,AE,AC,26,
    00,DD,6E,04,2ECC
112 DATA 9C80,22,A8,AC,DD,
    6E,02,22,AA,20C6
116 DATA 9CA0,B9,7E,E5,2A,
    AA,AC,45,C5,5AAF
118 DATA 9CB0,00,00,ED,5B,
    A8,AC,CD,F9,1E83
119 DATA 9CB8,BB,F1,C1,18,
    0B,ED,5B,97,7BAD
120 DATA 9CC0,B6,2A,A8,AC,
    19,22,97,B6,4F98
121 DATA 9CC8,10,E0,2A,99,
    B6,2B,2B,22,39B8
122 DATA 9CD0,99,B6,ED,5B,
    AC,AC,ED,53,7F49
123 DATA 9CD8,97,B6,C1,10,
    CA,E1,23,C1,7A73
124 DATA 9CE0,10,BB,ED,5B,
    A8,AC,21,08,396A
125 DATA 9CE8,00,CD,F0,9F,
    ED,5B,AC,AC,2340
126 DATA 9CF0,19,22,AC,AC,
    22,97,B6,2A,194A
127 DATA 9CF8,AE,AC,22,99,
    B6,E1,23,C1,7763
130 adr=&9C30:zeile=102:
MEMORY adr-1
142 SAVE"SCALE.BIN",B,
&9C30,&D5
```

Auch im Demoprogramm (Listing 2) sind einige Änderungen nötig:

```
10 adr=HIMEM-212:....
40 a=adr+34
50 POKE adr+27,INT(a/256):
POKE adr+26,a-256*INT
(a/256)
60 a=adr+30
70 POKE adr+21,INT(a/256):
POKE adr+20,a-256*INT
(a/256)
80 a=adr+26:POKE adr+18,
INT(a/256):POKE adr+17,
a-256*INT(a/256)
90 CALL adr+16:...
190 POKE &B72F,...
200 POKE &B72F,...
210 POKE &B72F,...
220 POKE &B72F,...
```

Zusätzlich ergänzen Sie dann noch die Zeile

```
82 POKE adr+187,INT(adr/25
6):POKE adr+186,adr-256*
INT(adr/256)
```

Damit steht der Anwendung des Befehls SCALE auf Ihrem CPC 6128 nichts mehr im Weg.

(Wolfgang Ried/ja)

Kybernetien in Not

Als Herrscher über den Staat Kybernetien (gleichnamiges Programmlisting aus dem 7. Schneider-Sonderheft, Seite 102) hatten viele Leser so ihre Probleme, denn was sie auch in die Erhaltung ihres Landes investieren wollten, der Computer meldete stets »zuviel«. Die Ursache dafür ist der immense Speicherplatzbedarf des Programms samt Variablen. Was auf einem CPC 464 mit Kassettenrecorder gerade noch so reicht, funktioniert auf Geräten mit angeschlossenem Diskettenlaufwerk nicht mehr. Die Abhilfe für diesen Umstand ist ebenso banal wie einfach. Starten Sie anstelle des Ladeprogramms (Listing 1) nur das Hauptprogramm (Listing 2) direkt. Was dadurch verlorengeht, sind lediglich die Kurzanleitung und die Erzeugung der deutschen Umlaute für den Bildschirmdialog. Das Spiel indes berührt diese Maßnahme nicht; bis auf die Tatsache, daß es nun korrekt läuft.

(Rolf Schultz/ja)

Tuning mit Hindernissen

Wie sich erst kurz nach dem Redaktionsschluß des 7. Schneider-Sonderhefts herausgestellt hat, verfügen einige Modelle des CPC 664 über eine andere Gate-Array-Version (Typ 40010) als die Mehrheit der Modelle 464 und 664.

Der Unterschied zwischen den Gate Arrays liegt glücklicherweise nicht in der internen Verschaltung, sondern lediglich in der Pinbelegung. Deshalb muß bei diesen Gate-Array-Versionen nicht die Leiterbahn an Pin 3, sondern die Leiterbahn an Pin 16 des Gate Array aufgetrennt werden. Das CAS-Signal
Signal für Pin 7 des PAL wird dementsprechend nicht an Pin 3, sondern an Pin 16 des Gate Array abgegriffen.

Ebenso müssen Sie das Signal ROMEN für den Umschalter US an Pin 27 statt an Pin 12 abgreifen.

Eine kleine Modifikation des Umbaus, die bei einigen Computertypen die Signalsicherheit erhöht und manchmal erst zur korrekten Funktion des Umbaus führt, besteht darin, daß Sie die Datensignale D6 und D7 für IC 74LS08 auf der PAL-Platine nicht an Pin 17 und Pin 18, sondern an Pin 16 (D6) und Pin 19 (D7) des IC 114 abnehmen. Da es – wie wir inzwischen feststellen mußten – allein vom CPC 464 mindestens fünf und vom CPC 664 mindestens drei verschiedene Versionen gibt, können wir Ihnen nicht pauschal mitteilen, für welchen Typ sich diese Änderung lohnt.

Im 7. Schneider-Sonderheft wurden im Schaltplan in Bild 1 auf Seite 16 zwei Signale vertauscht, so daß das Booten von CP/M Plus im 6128-Modus nicht korrekt funktioniert. Vertauschen Sie die Adreßleitungen A14 und A15 an den beiden PAL-Eingängen Pin 8 und Pin 9 miteinander, indem Sie A14 an Pin 9 und A15 an Pin 8 des PAL löten. Danach muß alles einwandfrei funktionieren.

Eine weitere Fehlerquelle, die eine korrekte Funktion nach dem Umbau verhindert, kann in der Verdrahtung liegen, wenn diese in Form von Kabelbäumen durchgeführt wird.

In diesem Fall liegen nämlich mehrere Signalleitungen so eng nebeneinander, daß es durch kapazitives und induktives Übersprechen der Signalspannungen zu Störungen im Spannungspegel benachbarter Signale kommt. Zudem sind durch die rechtwinklige Streckenführung der Kabelbäume die Signalleitungen länger als notwendig, was wiederum die Signalstärken schwächt.

Eine auf kurze Wege optimierte und aufgelockerte Führung der Leitungen schafft hier Abhilfe.

Probleme gibt es teilweise auch bei neuen CPC-Modellen, die mit einer Low-Power-Version des Gate Array ausgerüstet sind. Hier sind an den Ausgängen CAS und ROMEN die logischen TTL-Pegel für das PAL so knapp bemessen, daß sich der Einbau von je zwei Schmitt-Triggern (74LS14 oder 74LS19) in die CAS-Leitung zwischen Gate Array und PAL sowie in die ROMEN-Leitung zwischen Gate Array und Umschalter US lohnt, um die Signalpegel zu verbessern. Dadurch werden jedoch die genannten Signale um etwa 32 Nanosekunden verzögert, was wiederum zu Timing-Problemen beim Speicherzugriff führen kann.

Viele Leser haben uns angerufen und gefragt, ob der Umbau eines CPC auch mit der Vortex-Speichererweiterung funktioniert. Dazu kann folgendes gesagt werden:

Im 464-Modus arbeitet die Vortex-Speicherkarte im CPC einwandfrei, doch im 6128-Modus funktionieren sowohl Computer als auch Speicherkarte nicht. Dies liegt daran, daß einerseits das PAL des CPC zur Verwaltung der zweiten 64-KByte-Bank Bit 6 und Bit 7 der Portadresse 7Fxx (hex) auf 1 setzt, um auf andere Speicherbereiche zuzugreifen, und andererseits die Vortex-Speichererweiterung zum gleichen Zweck Bit 6 und Bit 7 der Portadresse 7Fxx (hex) auf 1 setzt. Dadurch kommt es zu Überschneidungen bei der Speicherverwaltung, die in einem sofortigen Absturz des Computersystems enden. Wir haben jedoch einen Hardware-Patch entwickelt, damit die Speichererweiterung auch im 6128-Modus einwandfrei läuft. Wenn Sie Interesse daran haben, dann wenden Sie sich telefonisch oder schriftlich an die Redaktion.

(Peter Bündgens/ma)

Tücken bei der Dekodierung

In die Dekodierung der Happy-Megabitkarte aus dem 7. Schneider-Sonderheft hat sich in Bild 5 und 6 leider ein kleiner Fehler eingeschlichen, der zu Fehlfunktionen der Speichererweiterung führt.

Bei dem Gatter, das das CS-Signal für den Datenbustreiber erzeugt und an den beiden Eingängen mit Pin 8 des IC 74LS27 sowie dem CE-Signal verbunden ist, darf es sich nicht, wie fälschlicherweise angegeben, um ein AND-Gatter 74LS08 handeln. Vielmehr muß hier ein OR-Gatter des IC 74LS32 eingesetzt werden. (Ein Gatter dieses Bausteins ist noch unbeschaltet.)

Das CE-Signal und das Signal von Pin 8 des IC 74LS27 müssen Sie an Pin 13 und Pin 12 des IC 74LS32 anschließen und das CS-Signal an Pin 11 des gleichen Gatters abnehmen.

Zusätzlich sind in Bild 6 Pin 4 und Pin 5 des IC 74LS27 an das WR-Signal angeschlossen. Diese beiden Anschlüsse müssen jedoch auf Masse gelegt werden. (Michael Stagge/ma)

Happy-Imager mit Partnerschaftsproblemen

Das Hardcopy-Programm »Happy-Imager« aus dem 7. Schneider Sonderheft (Seite 133) ist für Epson- und kompatible Drucker ausgelegt. Obwohl damit sein Betrieb mit den meistgebräuchlichen Druckern (auch dem DMP 2000) gewährleistet ist, gibt es natürlich auch Ausnahmen von dieser Regel. Wer die Hardcopyroutine nun an seinen speziellen Drucker anpassen möchte, erreicht dies durch einige wenige POKEs, deren Werte er aus seinem Druckerhandbuch entnimmt. Der erste wichtige Bereich ist die Initialisierung der Punktauflösung für den Grafikdruck. Wie Sie aus der Tabelle ersehen, geschieht diese Einstellung durch eine ganze Reihe von Bytes. Der Befehl selbst (»ESC *«) besteht aus den zwei Byte 27 (1B hex) und 42 (2A hex). Die 1 aktiviert eine Auflösung von 120 Punkten pro Zoll. Darauf folgen zwei Byte (Low- und High-Byte), die die Zahl der konkret zu druckenden Pixel bestimmen, in diesem Fall 800 (3*256+32).

Ab den Adressen 944B und 94B4 hex setzen je zwei Byte 1B und 40 hex (»ESC @«) den Drucker auf seine Standards zurück.

Adresse (hex)	Befehl (hex)	Wirkung
93EB	1B	ESC
93F0	2A	Grafikmodus
93F5	01	120 Punkte pro Zoll
93FA	20	Low-Byte der Punktzahl
93FF	03	High-Byte der Punktzahl
944B	1B	ESC
9450	40	Drucker-Initialisierung
94AF	0C	Seitenvorschub
94B4	1B	ESC
94B9	40	Drucker-Initialisierung
94D0	1B	ESC
94D5	33	n/216 Zeilenvorschub
94C3	0B	n=11 (0B hex)
94CB	01	n=1
94DD	0A	voller Zeilenvorschub
9665	1B	ESC
966A	21	Schriftartwahl
966F	1D	(für Copyright-Zeile)
9677	1B	ESC
967C	53	Super-/Subscript
9681	01	Subscript ein

Tabelle. Diese Druckerbefehle finden im Happy-Imager Verwendung.

Der Wert 0C hex bei Adresse 94AF hex sorgt für einen Seitenvorschub des eingelegten Papiers im Drucker.

Die Befehlsfolge 1B hex, 33 hex auf 94D0 hex bewirkt die Veränderung des Zeilenvorschubs (»ESC 3 n«). Ein dritter Wert steht als Parameter <n> für den neuen Wert des Vorschubs. Diesen erhält das Unterprogramm von zwei Routinen: Auf Adresse 94C3 hex sorgt das Byte 0B hex für 11/216 Zoll Vorschub, während die 1 (94CB hex) auf 1/216 Zoll schaltet.

Der Wert 0A hex auf 94DD hex veranlaßt einen vollen Zeilenvorschub.

Die Schriftart der Copyrightzeile bestimmt die Befehlsfolge ab Adresse 9665 hex (»ESC !«), wobei der dritte Wert (1D hex) für die Schriftart steht.

Dem Druck derselben Zeile dient der Befehl ab 9677 hex (»ESC S n«). Er schaltet in den Subscript-Modus, wenn wie hier der Parameter <n> eine 1 ist. Interessierte Leser können den gesamten Quellcode bei uns anfordern, wenn Sie uns eine formatierte Diskette im frankierten Rückumschlag senden. (Michael Herz/ja)

Impressum

Herausgeber: Carl-Franz von Quadt, Otmar Weber

Geschäftsführender Chefredakteur: Michael Scharfenberger

Chefredakteur: Michael Lang (lg)
Redakteure: Thomas Jacobi (ja, Projektleitung), Gregor Neumann (gn), Henrik Fisch (hf), Thomas Kaltenbach (kl), Martin Aschoff (ma);
Chef v. Dienst: Petra Wängler
Schlußredaktion: Eva Hierlmeier
Redaktionsassistenz: Monika Lewandowski (222), Rita Gietl (289)

Fotografie: Jens Jancke

Titelgestaltung: Katja Milles

Layout: Leo Eder (Ltg.), Katja Milles, Andrea Miller

Produktionsleiter: Klaus Buck (180)

Anzeigenverkaufsleitung: Ralph-Peter Rauchfuss

Auslandsrepräsentation:
Schweiz: Markt&Technik Vertriebs AG, Kollerstrasse 3, CH-6300 Zug, Tel. (042) 415656, Telex: 862329 mut ch
USA: M&T Publishing Inc., 501 Galveston Dr., Redwood City, CA 94063; Tel. 415-366-3600, Telex 752-351

Manuskripteinsendungen: Manuskripte und Programmlistings werden gerne von der Redaktion angenommen. Sie müssen frei sein von Rechten Dritter. Sollten sie auch an anderer Stelle zur Veröffentlichung oder gewerblichen Nutzung angeboten worden sein, muß dies angegeben werden. Mit der Einsendung von Manuskripten und Listings gibt der Verfasser die Zustimmung zum Abdruck in von der Markt&Technik Verlag AG herausgegebenen Publikationen und zur Vervielfältigung der Programmlistings auf Datenträger. Mit der Einsendung von Bauanleitungen gibt der Einsender die Zustimmung zum Abdruck in von Markt&Technik Verlag AG verlegten Publikationen und dazu, daß Markt&Technik Verlag AG Geräte und Bauteile nach der Bauanleitung herstellen läßt und vertreibt oder durch Dritte vertreiben läßt. Honorare nach Vereinbarung. Für unverlangt eingesandte Manuskripte und Listings wird keine Haftung übernommen.

Anzeigenverkauf: Britta Fiebig (211), Helmut Distl (398)

Anzeigenverwaltung und Disposition:
Patricia Schiede (172)

Marketingleiter: Hans Hörl (114)

Vertriebsleiter: Helmut Grünfeldt (189)

Vertrieb Handelsauflage: Inland (Groß-, Einzel- und Bahnhofsbuchhandel) sowie Österreich und Schweiz: Pegasus Buch- und Zeitschriften-Vertriebs GmbH, Hauptstätter Str. 96, 7000 Stuttgart 1, Tel. (0711) 6483-0

Bezugsmöglichkeiten: Leser-Service: Telefon (089) 4613-249. Bestellungen nimmt der Verlag oder jede Buchhandlung entgegen.

Bezugspreis: Das Einzelheft kostet DM 14,–

Druck: SOV St. Otto-Verlag GmbH, Laubanger 23, 8600 Bamberg

Verantwortlich:
Für redaktionellen Teil: Michael Lang
Für Anzeigen: Britta Fiebig

Redaktionsdirektor: Michael M. Pauly

Vorstand: Carl-Franz von Quadt, Otmar Weber

Anschrift für Verlag, Redaktion, Vertrieb, Anzeigenverwaltung und alle Verantwortlichen:
Markt&Technik Verlag Aktiengesellschaft, Hans-Pinsel-Straße 2, 8013 Haar bei München, Telefon (089) 4613-0, Telex 5-22052

Telefon-Durchwahl im Verlag:
Wählen Sie direkt: Per Durchwahl erreichen Sie alle Abteilungen direkt. Sie wählen (089) 4613 und dann die Nummer, die in Klammern hinter dem jeweiligen Namen angegeben ist.

ISSN 0931-5829

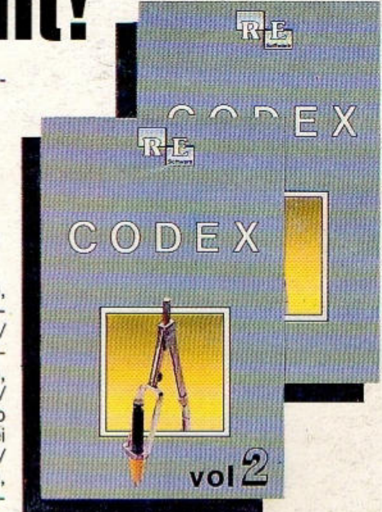
Irgendwann
kommt der Tag, an dem mit den Forderungen die Ansprüche steigen.
Dann sollten Sie vorbereitet sein. »PC Magazin« ist der entscheidende Schritt zur professionellen Computeranwendung.
PC Magazin
Die aktuelle Wochenzeitung für Personal Computer im IBM-Standard. ■ Wenn Sie an aktuellen und umfassenden Informationen über IBM-PCs und kompatible Systeme interessiert sind ■ Wenn Sie stets über die neuesten und effektivsten Anwendungen für den professionellen und privaten Bereich informiert sein wollen ■ Wenn Sie sich mit CAD/CAM und Netzwerken beschäftigen, dann ist das »PC Magazin« genau Ihre Zeitschrift.
Zur Anforderung Ihrer kostenlosen Probeexemplare einfach den nebenstehenden Gutschein ausfüllen, ausschneiden, auf eine Postkarte kleben oder in ein Kuvert stecken und einsenden an:
Markt & Technik, Verlag Aktiengesellschaft, PC Magazin Abonnenten-Service, Postfach 1304, 8013 Haar bei München.
PC Magazin
GUTSCHEIN
FÜR VIER KOSTENLOSE PROBEEXEMPLARE
Ich interessiere mich für »PC Magazin«, die Zeitschrift über IBM-PCs und Kompatible. Schicken Sie mir vier Ausgaben kostenlos als Probeexemplare.
Wenn ich »PC Magazin« nicht weiterlesen möchte, teile ich Ihnen dies sofort nach Erhalt der dritten Ausgabe mit. Gefällt mir »PC Magazin«, so daß ich es regelmäßig weiterbeziehen möchte, brauche ich nichts zu tun: Ich erhalte mein »PC Magazin« dann regelmäßig jede Woche per Post frei Haus geliefert und bezahle pro Jahr nur DM 155,— statt DM 229,50 im Einzelverkauf. Zustellung und Postgebühren übernimmt der Verlag.
Das Abonnement verlängert sich automatisch um ein weiteres Jahr zu den dann gültigen Bedingungen, wenn es nicht 2 Monate vor Ablauf schriftlich gekündigt wird.
Dieses Angebot gilt nur in der Bundesrepublik Deutschland einschließlich West-Berlin. Auslandspreise auf Anfrage.
Name
Vorname
Straße
PLZ
Ort
Datum, 1. Unterschrift
Mir ist bekannt, daß ich diese Bestellung innerhalb von 8 Tagen bei Markt & Technik, Hans-Pinsel-Str. 2, 8013 Haar widerrufen kann. Zur Wahrung der Frist genügt die rechtzeitige Absendung des Widerrufs. Ich bestätige dies durch meine 2. Unterschrift.
Datum, 2. Unterschrift
HCS18
PC Magazin
35
Viele Features einfach zu nutz
Preisknüller für AT-Einsteiger
ANSTELLUNGSVERTRAG

Über 1000 x verkauft!

Nach dem großen Erfolg von **"The Player's Dream"** und der Anwendersammlung **"CODEX"** kommen jetzt die Nachfolger:

The Player's Dream II
CODEX II

Für läppische DM 19.90 (Cass.) bzw. DM 24.90 (Disk.) finden Sie auf



"The Player's Dream I": Darts (12/85), Senso (12/85), Showdown (1/86), Jump Over (2/86), Pingo (2/86), Zentus (5/86), Steinschlag (6/86), Centibug (7/86), Jolly Jumper (8-9/86) und Pyramide (10/86).

CODEX I: Hexmonitor (12/85), Sprites mit Editor (12/85), Kalender (12/85), Datenverwaltung (1/86), DIR-Doctor (Directory-Editor) (2/86), Mini-Monitor (3/86), Mathematik (3/86), Statistik (4/86), Baudcopy (4/86), Hidump (Hardcopy) (4/86), Bücherdatei (5/86), Labelassembler ASSO (6/86), Notizblock (Sideklick) (6/86), Basic-Compiler (8-9/86), Disassembler (10/86).

"The Player's Dream II": Sepp im Hochhaus (4/86), Life (5/86), Minigolf (7/86), Tennis (11/86), Astronaut (12/86), Suicide Squad (2/87), Royal Flush (3/87), Flowers (3/87), Roulette (4/87) und Buggy Blaster (CK 10/85).

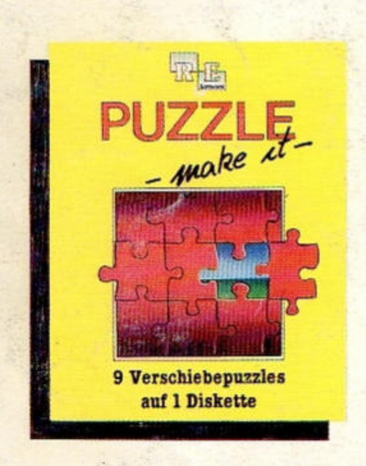
CODEX II: Softwareuhr (12/85), Disk-Doktor (1/86), Orgel (1/86), Datagenerator (2/86), Taschenrechner (3/86), Painter (3/86), Periodensystem (3/86), Elektro-CAD (5/86), Scrollbremse (6/86), Copy ??right!! V2.0 (6/86), 3-D-Prozessor (7/87), Digitalisierer (7/86), Tastenklick (8-9/86), Oszilloskop (8-9/86), Symbol-Editor (10/86), Fast-Routine (10/86), DFÜ (10/86), Datei (12/86), Neues HI-Dump (1/87).

Verschiebepuzzles

Wer kennt Sie nicht? Unser Mitarbeiter Andreas Zallmann und das Grafikass Christoph Schillo bearbeiteten 9x die Tasten ihres CPCs. Herausgekommen sind 9 "erlesene Köstlichkeiten". Puzzlen Sie mit!

9 Verschiebepuzzles auf 3"-Diskette für nur

DM 29.–



Die andere Software

10 Lernprogramme für Kinder zwischen 4 und 12 Jahren. Unser Mitarbeiter Berthold Freier, seines Zeichens Pädagoge aus Bayern, gestaltete 10 Programme, bei denen nicht Raketen und Gewehre im Vordergrund stehen, sondern der Grips der Kinder gefordert wird!

10 Lernprogramme auf 3"-Diskette für

DM 29.–

Software-Bestellschein

Ich bestelle aus dem CPC-Programmservice folgende Software:

Anzahl	Titel	Einzelpreis	Gesamtpreis
	Player's Dream I Cass.	19,90 DM	
	Player's Dream II Cass.	19,90 DM	
	Player's Dream I 3"-Disk	24,90 DM	
	Player's Dream II 3"-Disk	24,90 DM	
	CODEX I 3"-Disk	24,90 DM	
	CODEX II 3"-Disk	24,90 DM	
	Lernen mit Spaß 3"-Disk	29,00 DM	
	Puzzle 3"-Disk	29,00 DM	

Ich wünsche folgende Bezahlung:
- ☐ Nachnahme (zuz. 5.70 DM Versandkosten)
- ☐ Vorauskasse (keine Versandkosten)

Bei Vorauskasse bitte Scheck beilegen oder auf Postscheckkonto Karlsruhe 434 23-756 überweisen.

Name des Bestellers ______ PLZ/Ort ______

Anschrift ______ Datum/Unterschrift ______

Coupon ausschneiden, auf Postkarte kleben und einsenden an:
Verlag Rätz-Eberle/CPC-Software, Postfach 16 40, 7518 Bretten.